Scalable streaming learning of dyadic relationships

Thesis submitted for the degree of
"Doctor of Philosophy"

By Uri Shalit

Submitted to the Senate of the Hebrew University of Jerusalem

January 2015

This work was carried out under the supervision of:

Professor Daphna Weinshall and
Professor Gal Chechik

Acknowledgments

First, I wish to thank my advisors Gal Chechik and Daphna Weinshall. Without their guidance, support, knowledge and generosity this journey would not have been possible. I will miss the excitement of approaching them with a new idea, knowing they will always be interested, thoughtful, and encouraging. Daphna has been a constant positive force, sharing her knowledge and curiosity, always teaching me new ways to look at the problems that confronted me. I always left our meetings uplifted with new ideas and possibilities. Gal has guided me through these years, always showing me that I have more in myself that I believed. He showed me how to better myself and my work in ways I never thought possible. I want to thank him for sharing his knowledge, excitement, and seemingly endless capacity for turning things around from bad to good.

I wish to thank my colleagues and lab mates, for making this time so much more fun and interesting. Thank you to Noa, Lior, Ossnat, Hadas, Ronnie and Yuval from Gal's lab. Often times they were the best reason to go into the office in the morning, not to mention a source of support, inspiration, ideas, papers, and much needed distractions. I also want to thank my friends from the learning corridor in Jerusalem, always there, in good times and bad: Daniel, Roi, Alon, Elad and Elad made parts of this journey with me and made it much better that way.

I also wish to thank Professors Shai Shalev-Shwartz, Amir Globerson, Ami Wiesel, Gal Elidan and Yair Weiss. Their support, knowledge and wisdom were invaluable at crucial turning points in my work.

I wish to thank Google for their generous support through the Google Europe Machine Learning Fellowship, and especially Beate List from Google for turning Google's support into a true gift.

Finally, my deepest gratitude and love go to my family. My wife Tali who made all this possible - words are not enough to describe my debt of love. My parents who implanted in me curiosity, love of knowledge, and the joy of discovery. My daughters Shira and Rotem - the best thing that ever happened to me.

Abstract

Modern machine learning problems are often about learning a mapping between two high-dimensional domains. For example, learning a mapping from the domain of images to the domain of words, or learning which documents are similar to others. We refer to these mappings as *dyadic relationships*. In this thesis we focus on learning dyadic relationships encoded in matrix form. For example, a matrix mapping between image representations and semantic representations, or a matrix acting as a bilinear form encoding similarity between pairs of documents.

Specifically, we address the challenge of learning dyadic relationships for large-scale high-dimensional datasets. We employ two sets of tools in developing the methods presented here: First, we use the rich geometric and algebraic structure of matrices. This includes the Riemannian manifold structure of the set of low-rank matrices, decompositions of orthogonal matrices, and properties of positive definite matrices. In addition, we concentrate on streaming access algorithms. These are algorithms which have access to a small subset of the data samples or features at any one time. Streaming algorithms are key for handling huge datasets which do not fit in memory, and for obtaining fast predictions even before the entire data is available or processed.

This thesis is based on four papers. First, we deal with the problem of learning a similarity measure between sparse high-dimensional instances, applied to image retrieval. We focus on building a fast algorithm which scales to large datasets, and investigate the role of symmetry and positive definiteness in its performance. The second paper presents two streaming algorithms for learning low-rank and low-rank positive definite matrices, applied to learning similarity measures and multi-label models. The algorithms are based on Riemannian stochastic gradient descent, and prove to be both fast and robust despite the highly non-convex nature of the low-rank constraint. The third paper deals with the challenge of learning interpretable relations - how can an algorithm explain its output in terms which make sense to humans? The work is motivated by the need to understand a large dataset of high-resolution gene expression images in the mouse brain. The images exhibit elaborate multi-scale patterns, and the genes themselves exhibit complex interactions, making the data difficult to comprehend. Our method uses the semantic information gained from the rich existing knowledge on genes and their functions, and gives researchers means for exploring and understanding the dataset. Finally, the fourth paper deals with the general problem of efficiently learning orthogonal matrices, motivated by recent applications to tensor decompositions for method-of-moments estimation of latent variable models. We present a novel Riemannian coordinate descent algorithm, based on simple sparse orthogonal matrices called Givens rotations. We show our method produces better results for tensor decomposition compared with the state-of-the-art, as well as being faster and outperforming a state-of-the-art algorithm for sparse PCA.

We conclude by discussing several future research directions stemming from the work presented here, as well as building on recent advancements by other researchers.

Letter of contribution to chapters with joint authors

Contributions to Chapter 2.1

- Jointly with Gal Chechik, designed and carried out the Caltech256 experiment (section 5.3).
- Jointly with Gal Chechik, designed and carried out the parallel training experiment (section 5.4).
- Jointly with Gal Chechik, designed and carried out the symmetry and positivity experiment (section 6).
- Jointly with Gal Chechik, analyzed the above experiments and wrote corresponding sections 5.3, 5.4 and 6.
- Performed and wrote the theoretical loss bound analysis (section 2.2).
- Jointly with Gal Chechik and Samy Bengio, wrote the related work section.
- Jointly with Gal Chechik, Samy Bengio and Varun Sharma, wrote the discussion (section 7).

Contributions to Chapter 2.3

- Jointly with Noa Liscovitch, conceived and designed the project.
- Designed the machine learning model presented and used in the paper.
- Designed the similarity measures and interpretable similarity concept presented and used in the paper.
- Jointly with Noa Liscovitch and Gal Chechik, designed and carried out the experiments and their analysis.
- Wrote the paper jointly with Noa Liscovitch and Gal Chechik, was the lead writer of sections 2.2 and 2.3 and prepared figures 2, 4 and S2.
- Jointly with Noa Liscovitch and Gal Chechik, wrote Introduction and Summary.

Contents

Chapter 1

Introduction

Numerous machine learning problems involve a mapping between two high-dimensional domains. As a common example, multi-class learning can be viewed as learning a mapping from the instance domain to the label domain. As another example, learning a similarity measure can be viewed as learning a mapping between a domain to itself. Collaborative filtering can be viewed as learning a mapping between the user domain and recommended item domain. We call these *dyadic relationships*.

In fact, one can argue that the majority of real-life machine learning problems are dyadic or even polyadic. Computer vision tasks typically involve many labels and objects, a typical biological application can involve inferring relationships pertaining to thousands of genes and hundreds or thousands of biological conditions, and learning for ad placements demands learning a relationship between thousands of ads and millions of possible queries.

A classic mathematical representation for a linear mapping between two high-dimensional domains is the *matrix*. Using some of the examples above, multi-class learning can be seen as learning a matrix of classifiers; similarity learning as learning a bilinear form parametrized by a matrix; and collaborative filtering as completing a partially observed user-item interaction matrix.

Formulating dyadic problems as matrix problems assumes a flat structure on each of the domains. Recently there has been much interest in learning structured dyadic relationships, for example using the structure of language when learning a model for annotating images (Frome et al., 2013). Adding complex structure to the domains of the dyadic relationships is beyond the scope of this thesis, and will be considered in future work.

Learning mappings between high-dimensional spaces is becoming ever more challenging in recent years as the size, richness and requirements from machine learning algorithms have soared. For example, the number of available labels for images have grown to the tens of millions, as did the numbers of users and items in item recommendation tasks.

In addition, the recent explosion in the size of available data has outpaced the growth rate of computing power and local storage capacity. This has made many classical machine learning methods such as kernel methods increasingly irrelevant, and brought a rising interest in developing new algorithms suited for these new conditions

This work aims to create methods for learning dyadic relationships suited for handling large-scale datasets: Ideally, the runtime and memory demands of our methods scale linearly or sublinearly with the number of instances and with the dimensionality, the methods have low communication cost, and they are amenable for parallelization.

We combine two sets of tools to approach these goals: The first set of tools is harnessing the rich algebraic and geometric structure of matrix domains. For example, the many characterizations of matrix rank provide a fertile ground for creating efficient learning algorithms. As another example, we use the Riemannian manifold structure of sets such as low-rank positive semidefinite matrices

or orthogonal matrices to create efficient and scalable algorithms for learning dyadic relationships.

The second set of tools is using a simple yet attractive family of optimization methods particularly well suited for large datasets: *streaming access algorithms*. In the streaming access model, only a small subset of the data is available to the algorithm at any one time. A streaming data access model enables a small memory footprint, since only a small fraction of a huge dataset is used for any given computation. In their 2008 paper "The Tradeoffs of Large-scale Learning", Bousquet and Bottou (2008) argued persuasively in favor of using simple streaming access optimization methods in the large sample regime. They show from a theoretical standpoint why simple streaming methods such as stochastic gradient descent give better generalization when compared with first order gradient or second order Newton methods which require processing an entire dataset.

An additional important advantage of the streaming data optimization model is handling learning tasks where the data actually arrives in a stream: for example the task of recommending daily news items to a user. In such cases the algorithm needs to deliver predictions while the data is being gathered.

This thesis is organized as follows. Chapters 2.1 and 2.2 build on the basic concept of stochastic gradient descent, where the data access is to a few instances at a time. We first use sparsity, then the Riemannian geometry of low-rank matrices, to develop fast and efficient algorithms for learning dyadic relationships expressed in matrix form. We apply these new algorithms to large-scale learning of image and document similarity, and to image labelling. In Chapter 2.3 we take a slight detour and focus on a specific challenge: how to learn similarity models that can be interpreted by humans? We approach this problem motivated by the need to help scientists understand a newly available, complex dataset of gene expression maps in the mouse brain. Chapter 2.4 of this thesis develops a coordinate descent approach for Riemannian space, where the data access is to a few features at a time. The Riemannian coordinate descent method leads in turn to a new approach for learning orthogonal matrices and to a novel tensor decomposition method.

In this introduction, we first discuss the applications of learning dyadic relationships in machine learning. Then we overview some of the basic geometric and algebraic matrix structures used throughout this thesis, such as positive definiteness and low-rank. Finally, we introduce some basic tools of optimization, and specifically Riemannian optimization, which have proven so fruitful in the research presented here.

1.1 Applications of learning dyadic relationships with matrices

This section outlines some motivating examples of applications of learning dyadic relationships in machine learning, as discussed and applied throughout this thesis.

1.1.1 Metric learning and similarity learning

Many machine learning tasks inherently rely on the existence of a metric or a similarity measure between pairs of instances. For example, retrieving a document similar to a query document, nearest-neighbor classification, recommending a similar item, and clustering, all assume that a way to measure distances between instances exists.

The idea of *similarity learning* is that this similarity measure or metric can itself be the object of machine learning. It has been shown in many cases that learning the similarity measure can dramatically improve the performance of machine learning algorithms (Xing et al., 2002; Bar-Hillel et al., 2005; Davis et al., 2007; Jain et al., 2008; Weinberger and Saul, 2009; Guillaumin et al., 2009b,a; Kulis, 2012).

For example, learning a metric and then using it for nearest neighbor classification has been used to obtain (at the time) state-of-the-art results in face recognition tasks (Guillaumin et al., 2009b) and to improve human activity recognition in video (Tran and Sorokin, 2008). As another example, consider the problem of "query-by-example", when the user wishes to find images or documents similar to a query document; in Chapter 2.1 of this thesis we learn a similarity measure that makes for much better query-by-example retrieval when compared with similarity measures such as the oft-used cosine similarity measure (Manning et al., 2008).

The metrics most commonly used in machine learning are **Mahalanobis distances**: For a pair of vectors $x, y \in \mathbb{R}^n$, the Mahalanobis distance between them is parametrized by a matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$, and given by:

$$dist_{\mathbf{M}}(x, y) = \sqrt{(x - y)^T \mathbf{M}(x - y)}. \tag{1.1}$$

For $\mathbf{M} = I$ the identity matrix, the Mahalanobis distance is identical to the standard Euclidean distance.

In order for the function $dist_{\mathbf{M}}$ to be a true metric, it must be (1) symmetric, (2) positive on distinct pairs $x \neq y$, and (3) obey the triangle inequality. These three conditions are fulfilled if and only if the matrix $\mathbf{M}$ is *positive definite* (PD). That means that $\mathbf{M}$ must be symmetric with strictly positive eigenvalues (see Definition 1 in Subsection 1.2.1 below). If $\mathbf{M}$ is positive *semidefinite* (PSD), that is some of its eigenvalues are zero, then the symmetry and triangle inequality conditions hold, but pairs of vectors might have zero distance despite being distinct. More precisely, any pair of vectors $x, y \in \mathbb{R}^n$ whose difference $x - y$ lies in the null space of $\mathbf{M}$ will have Mahalanobis distance of 0. For PD matrices the null space is $\{0\}$, meaning only identical vectors have distance 0. See further discussion on PD and PSD matrices in Subsection 1.2.1 below.

Alternatively, instead of learning a distance metric we can learn a similarity measure parametrized by a matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$. The similarity measure is given by:

$$sim_{\mathbf{S}}(x, y) = x^T \mathbf{S} y. \tag{1.2}$$

If $\mathbf{S} = I$ the identity matrix, then $sim_{\mathbf{S}}$ is simply the standard inner product. For $sim_{\mathbf{S}}$ to be an inner product, it must be symmetric and obey $sim_{\mathbf{S}}(x, x) \geq 0$, $sim_{\mathbf{S}}(x, x) = 0 \Leftrightarrow x = 0$. In matrix terms, symmetry is achieved if and only if the matrix $\mathbf{S}$ is symmetric, and positivity if and only if $\mathbf{S}$ is positive definite. Similar to the Mahalanobis case, if $\mathbf{S}$ is only positive semidefinite, then some non-zero vectors will have $sim_{\mathbf{S}}(x, x) = 0$: exactly the vectors x in the null space of $\mathbf{S}$.

In Chapter 2.1 of this thesis we introduce a method for learning a similarity measure $\mathbf{S}$, called OASIS. OASIS is based on the Passive-Aggressive algorithm (Crammer et al., 2006), which is a close variant of SGD. Using this variant of SGD, along with leveraging sparse instance representations, makes OASIS an extremely fast method for learning similarity. We applied OASIS to the challenging task of learning semantic image similarity, for example, learning that two images of airplanes are similar even when one is on the ground and one is in the air, while an image of an airplane and a rainbow are not as similar despite both of them showing the sky. OASIS achieved (then) state-of-the-art results on the Caltech256 dataset (Griffin et al., 2007).

However, OASIS suffered from two caveats. The first caveat is that there is no computationally cheap way to make the similarity positive definite. The most straightforward way would be projecting onto the set of PSD matrices. However, this projection is computationally intensive, since it is based on an eigendecomposition, requiring computation time which is cubic in the instance dimension. The second caveat is that the model size scales quadratically with the instance dimension, even when the instances themselves are sparse.

In order to address these two issues, we introduce in Chapter 2.2 a new algorithm called

LORETA. LORETA learns a low-rank similarity matrix directly, which yields significant computational and memory benefits. Particularly, if the instance domain is of dimension n and the similarity matrix is of rank r, the memory needed for storing the matrix is $O(nr)$ instead of $O(n^2)$. We discuss this point further in Subsection 1.2.2 below. In addition, LORETA is capable of directly learning a low-rank PSD matrix, without the computational burden of an eigendecomposition which was needed in our previous work on OASIS. The major innovation introduced in LORETA is a way to perform computationally efficient SGD within either the manifold of low-rank matrices or low-rank PSD matrices. Using the manifold structure enables the creation of an efficient and stable algorithm.

We note that both OASIS and LORETA learn bilinear models, while the non-linear elements such as image features were precomputed. However, bilinear models can be readily trained as part of a deeper, nonlinear learner - see for example systems suggested by Zhong et al. (2011); Wu et al. (2013).

1.1.2 Multiple label learning

Consider the problem of object recognition in images. Suppose we have a large list of possible objects, L. We wish to have an algorithm that could assign to a given image several object labels from L, possibly sorted by prominence or confidence. A straightforward way to approach this problem is to treat it as a collection of $|L|$ binary classification problems, one for each object label $l \in L$. This method has been used successfully, for example by Crammer and Singer (2003); Kakade et al. (2008) and more recently has garnered much success in the computer vision community, for example by Torresani et al. (2010); Li et al. (2010); Deng et al. (2011).

However, in many cases it makes sense to tie the different labels together. For example, if the task is image labelling, then the classifiers for mammals such as bears, dogs and sheep might be more similar, or use similar features, when compared with classifiers for sky-scrapers or guitars. Assume there exists a *latent* label space S of size $|S| \ll |L|$, such that the labels of L are combinations of the latent labels S (Amit et al., 2007).

A matrix formulation of this idea is as follows. Let d be the instance dimension, and let $B \in \mathbb{R}^{|S| \times d}$ be a matrix of classifiers for the latent image space. Assume that the classifiers for the labels $l \in L$ are linear combinations of the latent label classifiers, and let $A \in \mathbb{R}^{|L| \times |S|}$ be a matrix encoding the corresponding linear combinations. We then form a classifier matrix $C = AB \in \mathbb{R}^{|L| \times n}$, such that for an instance $x \in \mathbb{R}^n$, the product Cx gives a score for each possible label $l \in L$. Since the number of latent labels is much smaller than the total number of labels, the matrix C is of *low-rank*. Thus, we have reduced the problem of learning latent label spaces with linear combinations to the problem of learning a low-rank classifier matrix C.

In Chapter 2.2 we use this insight and learn a low-rank multi-label matrix model, applying it to label images from the ImageNet dataset (Deng et al., 2009). The labels for these images form a hierarchy, with strong correlations between certain label subsets. We show that learning low-rank multi-label models can significantly improve performance over full-rank models, while saving memory and computation resources.

1.1.3 Interpretable similarity learning

A challenge faced by many machine learning applications, especially when used in scientific context, is the difficulty of interpreting the learned model. For example, a scientist working on understanding relations between different genes has only limited use for a machine learning similarity model that simply states that gene A and gene B are “similar”. Researchers often wish for more insight into the factors underlying the predicted similarities. This stands in contrast to machine learning tasks such as object-recognition, where the goal is to replicate a well developed human ability. In

scientific machine learning applications we often wish to gain a deeper understanding of relations which are difficult even for human experts to observe.

In Chapter 2.3 we tackle the challenge of creating an interpretable similarity model in the context of bioimaging. The specific problem motivating us is making sense of a new, large set of high-resolution gene expression images in mammalian brains made available by the Allen Institute for Brain Science (Lein et al., 2006; Ng et al., 2009; Hawrylycz et al., 2014). These images convey the level of expression of each of 20,000 genes on a sub-cellular resolution within mouse brains. Our aim is to create an algorithm which makes it easier for neuroscientists to understand the relations between genes, their biological functions, and the genes' expression patterns in the brain.

To this end we use a two-stage approach leveraging a rich human-curated knowledge base about genes and their biological functions called the Gene Ontology (Gene Ontology Consortium, 2004). The Gene Ontology (GO) is a directed acyclic graph (DAG) with nodes corresponding to biological function, ranging from very general functions at the head of the DAG, such as "metabolism" to very specific functions such as "negative regulation of systemic arterial blood pressure". Each of the functions is annotated with genes that have been shown to relate to that function, with genes often annotated with multiple biological functions.

The first stage of our interpretable similarity learning approach is training a probabilistic classifier for each function in the Gene Ontology, aiming to predict biological function from a brain gene expression image. We then treat the outputs of these classifiers as features. Since the Gene Ontology is created and curated by humans, using the presence of GO functions as features creates inherently interpretable features. In the second stage we further use the GO structure for two purposes: evaluating similarities between genes based on their expression patterns, and explaining similarities between genes in terms of GO functions. For example, two genes that share many functions that are close within the GO DAG structure would be deemed similar. These functions would also be the explanatory factors proposed by the algorithm.

Explaining the similarities opens the possibility for finding distinct types of similarity. For example, our algorithm can suggest that gene A is similar to gene B because both A and B are related to a certain neurotransmitter, and that A is similar to gene C because both A and C relate to a specific metabolic mechanism. In Chapter 2.3 we describe our interpretable similarity model in detail, and show its use in gaining new insights into the functions and relations between genes. Sections 3.3 and 3.4 of Chapter 2.3 give a detailed example where our model predicts several similarities to a gene called Synpo2, while offering distinct and diverse explanations for these similarities.

1.2 Matrix structures

1.2.1 Positive definiteness

Matrix algebras are in general more intricate than the algebra of the real numbers, even if we restrict ourselves to the set of square symmetric matrices. For example, the set of square symmetric matrices isn't well ordered. However, there is a natural matrix generalization to the idea of a positive real number - the positive definite (PD) matrix. Similar to a real number, every PD matrix admits a real matrix square root. This feature is of particular importance when using matrices to parametrize distances and similarities.

Definition 1. *A square symmetric matrix* $A \in \mathbb{R}^{n \times n}$ *is* positive definite (PD) *if all its eigenvalues are strictly greater than* 0*, and* positive semi-definite (PSD) *if all its eigenvalues are greater than or equal to* 0*.*

Corollary 1. *For every PD or PSD matrix* A *there exists a real matrix* B *such that* $A = BB^T$*.*

Proof. Let $A = U\Lambda U^T$ be the spectral decomposition of A, where $U\mathbb{R}^{n\times n}$ is an orthogonal matrix such that $UU^T = U^TU = I_n$, and Λ is a real diagonal matrix. This decomposition exists since A is symmetric. From Definition 1, all the diagonal elements of Λ are non-negative. Let $B = U\sqrt{\Lambda}V^T$, where the square-root of Λ is element-wise, and V is *any* $n \times n$ orthogonal matrix. Then $A = BB^T$. □

If the matrix A is positive semi-definite, then necessarily A has rank $k \leq n$, since A could have 0 eigenvalues. In that case, the matrix B can be set to be of dimensions $n \times k$, spanning only the eigenspace of the non-zero eigenvalues of A.

In both the PD and PSD cases, the existence of a decomposition as in Corollary 1 implies that the metric $dist_{\mathbf{M}}$ (Eq. 1.1), or the similarity $sim_{\mathbf{S}}$ (Eq. 1.2) in fact rely on a linear transformation of Euclidean space. Let $M = BB^T$, with $B \in \mathbb{R}^{n\times k}$, $k \leq n$:

$$dist_{\mathbf{M}}(x, y) = \sqrt{(x-y)^T\mathbf{M}(x-y)} = \sqrt{(x-y)^TBB^T(x-y)} = dist_{I_k}(Bx, By), \tag{1.3}$$

where $dist_{I_k}$ is simply the Euclidean distance in R^k (if M is positive definite then $k = n$). Similarly, let $\mathbf{S} = BB^T$ with $B \in \mathbb{R}^{n\times k}$, $k \leq n$:

$$sim_{\mathbf{S}}(x, y) = x^T\mathbf{S}y = x^TBB^Ty = sim_{I_k}(Bx, By).$$

We see that positive definite and positive semi definite distance and similarity measures can be understood as follows: apply a linear transformation B to the data, and measure distances or similarities in the transformed space using the standard Euclidean distance or the standard inner product. If the distance or similarity is positive semi-definite and not positive definite, then B is dimension reducing.

Chapters 2.1 and 2.2 of this thesis explore extensively these ideas. In Chapter 2.1 we introduce an algorithm for learning a general (not necessarily PSD) similarity matrix $\mathbf{S}$. In Section 6 of Chapter 2.1 we examine the effect of enforcing the PSD attribute on the learned matrix $\mathbf{S}$, by projecting $\mathbf{S}$ onto the set of PSD matrices. In general we find that requiring $\mathbf{S}$ to be PSD improves the quality of the learned similarity measure. This implies that requiring the similarity measure to rely on an actual Euclidean representation serves as form of prior or regularization for similarity learning. This was further shown in the work of Qian et al. (2013).

A drawback in enforcing the PSD attribute by projection onto the PSD set is that it is computationally expensive, requiring repeated eigendecompositions to identify the space of negative eigenvalues. In Chapter 2.2 we explore a new approach allowing us to directly learn a similarity measure parametrized by a PSD matrix. This is done by using the Riemannian geometry of the set of PSD matrices and optimizing directly over the PSD manifold, avoiding expensive eigendecompositions altogether.

1.2.2 Low rank

Matrix models in machine learning are often very high-dimensional. For example, the number of parameters in a matrix similarity model in $\mathbb{R}^n$ is n^2; a full user-item preference matrix model for collaborative filtering is of size $\#items \times \#users$, which can be huge - far larger than can be stored in the memory of a single or even a few modern machines.

Given this potential for very high dimensionality, we are often in search for ways to limit the number of parameters. Limiting the number of parameters serves two causes: First, it leads to better generalization, or less overfitting. Second, it makes the model tractable in terms of memory and computation time.

An extremely useful way to limit the number of parameters in matrix models is imposing a

low-rank constraint. A matrix $X \in \mathbb{R}^{n\times m}$ is of rank k if it has a factorization $X = AB^T$, where $A \in \mathbb{R}^{n\times k}$ and $B \in \mathbb{R}^{m\times k}$ are "thin" matrices. The overall number of parameters for a low-rank X is reduced from $n \cdot m$ to $(n+m) \cdot k$, which is much smaller if $k \ll min(n, m)$.

For the special case of a positive semi-definite matrix X, it is straightforward to show that any rank-k PSD matrix $X \in \mathbb{R}^{n\times n}$ can be factored as $X = AA^T$ with $A \in \mathbb{R}^{n\times k}$. See the discussion in Subsection 1.2.1 above.

Constraining a matrix to be low-rank can have many interpretations according to the learning scenario. In the case of similarity learning, we have for a low-rank similarity matrix $S = AB^T$

$$sim_{\mathbf{S}}(x, y) = x^T \mathbf{S} y = x^T AB^T y = sim_{I_k}(Ax, By). \tag{1.4}$$

This means that the matrices A and B project the vectors x and y respectively from n dimensions to a small k-dimensional space, and in that low-dimensional space similarity is given by the standard inner-product. If S is PSD, then we can factor $S = AA^T$, meaning the projections of the left vector x and right vector y are identical.

While a low-rank decomposition has clear advantages, it poses difficulties from an optimization point of view. The set of low-rank matrices is highly non-convex, and in general minimizing a convex function subject to a rank constraint is NP-hard (Natarajan, 1995). Even finding good local minima is often hard due to the fact that the decomposition $X = AB^T$ is not unique and has a large invariant space. Let $M \in \mathbb{R}^{k\times k}$ be any invertible matrix. Then we can replace the matrix A by AM and the matrix B by BM^{-T}, and obtain $X = AM\left(BM^{-T}\right)^T = AB^T$. This non-uniqueness makes optimization unstable.

In Chapter 2.2 of this thesis we use the Riemannian manifold structure of the set of low-rank matrices to resolve these difficulties. We develop an algorithm, LORETA, for numerically stable learning of low-rank matrices with computational and memory burden which is linear in the number of parameters of the low-rank factorization.

1.2.3 The structure of orthogonal decompositions

The idea of orthogonal matrix decompositions is extremely powerful and widely used in mathematics and applications. The classic matrix spectral theorem guarantees that for any symmetric matrix $A \in \mathbb{R}^{n\times n}$ such that $A = A^T$ there exists an orthogonal matrix $U \in \mathbb{R}^{n\times n}$ and a diagonal matrix $\Lambda \in \mathbb{R}^{n\times n}$ such that $UU^T = U^TU = I_n$ (orthogonality) and $A = U\Lambda U^T$. If the eigenvalues of A are distinct, then the matrix U contains the eigenvectors of A, and the matrix Λ the corresponding eigenvalues.

More generally, the singular value decomposition for any (not necessarily square) matrix $B \in \mathbb{R}^{n\times m}$ is $B = USV^T$ with $U \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{m\times m}$ both orthogonal matrices, and $S \in \mathbb{R}^{n\times m}$ a matrix with non-negative entries on its main diagonal, and 0 otherwise.

In machine learning applications, the use of such orthogonal decompositions as the eigendecomposition and SVD is ubiquitous: Principal Component Analysis (PCA), Latent Semantic Analysis (Dumais, 2004), dimension reduction via SVD, and matrix completion with SVD (Jain et al., 2010; Koren et al., 2009) are some of the better known examples.

Looking into constructing efficient streaming algorithms has lead us to look into how these orthogonal decompositions are calculated in practice (Golub and Van Loan, 2012). In most cases, at the core of the calculation lies one of two very simple orthogonal matrices: *Householder reflections*, or *Givens rotations*.

A Householder reflection is generated by a normalized vector $v \in \mathbb{R}^n$ and has the form $H = I - 2vv^T$. H is symmetric and orthogonal, and its determinant is -1.

A Givens rotation is an orthogonal matrix of the form:

$$G(i,j,\theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & cos(\theta) & \cdots & -sin(\theta) & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & sin(\theta) & \cdots & cos(\theta) & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix},$$

where the entries $cos(\theta)$ and $sin(\theta)$ are on the (i,j) submatrix of $G(i,j,\theta)$.

Any orthogonal matrix $U \in \mathbb{R}^{n\times n}$ with $det(U) = 1$ can be represented as a product of $\frac{n(n-1)}{2}$ Givens rotations. Any orthogonal matrix can be represented as a product of n Householder reflections and one Givens rotation.

The simplicity of both Householder and Givens orthogonal operators make them ideal for efficient optimization. In Chapter 2.4 of this thesis we formally show that the application of Givens rotations is in fact a form of coordinate descent in the Riemannian manifold of orthogonal matrices. We use this insight to derive a general, efficient, and parallelizable CD procedure for learning orthogonal matrices. We apply this method to two problems: first is learning a sparse PCA model, following the formulation of Journée et al. (2010). Second, we give a new method for orthogonal tensor decomposition, an optimization problem which has risen to prominence lately as a means for performing a method-of-moments estimation for statistical models such as Gaussian Mixture Models and Latent Dirichlet Allocation (Anandkumar et al., 2012a). We show our new method is consistently more robust than the current state of the art, the Tensor Power Method of Anandkumar et al. (2012a, 2013).

1.3 Large-scale matrix optimization

Modern data sets are usually large in two different aspects: they include many instances, and the instances themselves are high-dimensional. For example, the Wikipedia online encyclopedia includes both millions of documents (instances), and each document is comprised of a vocabulary containing millions of words, as well as a highly elaborate link and tag structure. The streaming methods in this thesis adopt one of two approaches: either update the model using a few instances at a time, or a few dimensions at a time.

These two approaches correspond to two very well known optimization methods: For the case of updating a model using a few instances at a time, a simple and extremely popular algorithm is Stochastic Gradient Descent (SGD). The case of updating using a few dimensions at a time corresponds to a family of methods known as Coordinate Descent (CD).

There is a vast and ongoing research effort into understanding the strengths and weaknesses of both SGD and CD. A pioneering work analyzing SGD in the framework of online learning is Bottou (1998). For more current tools and ideas regarding the use of SGD in machine learning, see work by Zhang (2004); Bottou (2010). For newer ideas regarding parallelizing SGD, see Recht et al. (2011). For newer analysis regarding SGD for smooth and strongly convex functions, as well as finite-sample guarantees, see Rakhlin et al. (2011); Shamir and Zhang (2012); Needell et al. (2013). Finally, for a discussion of SGD on Riemannian manifolds (post-dating the relevant work in this thesis) see Bonnabel (2013). Regarding coordinate descent, it has been studied extensively inside the machine learning and statistics communities as a means to solve the dual SVM problem (Joachims, 1999; Hsieh et al., 2008) and the LASSO and related non-smooth problems (Wu and Lange, 2008; Friedman et al., 2010; Shalev-Shwartz and Tewari, 2011; Takáč et al., 2013). A better

understanding and stronger convergence guarantees of CD, and specifically randomized CD have been given in papers by Nesterov (2012); Richtárik and Takáč (2014); parallel CD is discussed by Richtárik and Takáč (2012), and a method to perform proximal, non-smooth, and parallel CD is given by Fercoq and Richtárik (2013).

Chapters 2.1 and 2.2 of this thesis build on the concept of SGD. In chapter 2.1 we leverage sparse representations and the Passive-Aggressive algorithm (Crammer et al., 2006) which is a variant of SGD, to obtain an extremely fast algorithm for learning a similarity measure between images. In Chapter 2.2 we develop the first Riemannian SGD scheme in the manifold of low-rank matrices and low-rank PSD matrices, creating a fast, stable, and low-memory algorithm for learning of low-rank matrices. We apply this algorithm to learning of similarity between documents, and to rank labels for images from the ImageNet dataset Deng et al. (2009). In Chapter 2.4 we develop a coordinate descent approach for the manifold of orthogonal matrices, using the simple and sparse Givens rotations (see Subsection 1.2.3 above) as the basic building block.

1.4 Manifolds and Riemannian optimization

In this subsection we outline the general principles of optimization over Riemannian manifolds, which are used extensively in Chapters 2.2 and 2.4 of this thesis. The standard reference for this subject is the book "Optimization algorithms on matrix manifolds" by Absil et al. (2009).

Riemannian optimization has emerged in recent years as an exciting tool for solving difficult, highly non-linear optimization problems (Edelman et al., 1998; Keshavan et al., 2009; Absil et al., 2009; Journée et al., 2010; Turaga et al., 2011). In this thesis we use the Riemannian manifold structure of the set of fixed-low-rank matrices to come up with novel, efficient and stable Riemannian stochastic gradient learning algorithms, as detailed in Chapter 2.2. In Chapter 2.4 we use the Riemannian manifold structure of the orthogonal matrix group to develop a Riemannian coordinate descent procedure based on successive multiplication of Givens rotations.

An *embedded manifold* is a smooth subset of an ambient space $\mathbb{R}^n$, locally homeomorphic to $\mathbb{R}^d$, with $d \leq n$ being the manifold dimension (Do Carmo, 1992). For instance the set $\mathcal{S}^{n-1} = \{x : \|x\|_2 = 1, x \in \mathbb{R}^n\}$, the unit sphere, is an $n-1$ dimensional manifold embedded in n-dimensional Euclidean space $\mathbb{R}^n$. As another example, the *orthogonal group* O_n, which comprises of the set of orthogonal $n \times n$ matrices, is an $\frac{n(n-1)}{2}$ dimensional manifold embedded in $\mathbb{R}^{n \times n}$. A third example is the manifold of *low-rank* matrices, the set of $n \times m$ matrices of rank k where $k < min(m, n)$. This set is an $(n+m)k-k^2$ dimensional manifold embedded in $\mathbb{R}^{n \times m}$. Embedded manifolds inherit many properties from the ambient space, a fact which simplifies their analysis. For example, a canonical Riemannian metric for embedded manifolds is simply the Euclidean metric restricted to the manifold.

The goal of Riemannian optimization is to minimize a loss function $L(W)$ under the constraint that W is a member of a manifold $\mathcal{M}$.

Let us consider now the elements of a very basic optimization procedure - gradient descent (GD). In order to perform GD in Euclidean space, one obtains the gradient of the function, and takes a step in that direction. In order to perform this in Riemannian space, two issues must be addressed: first, define a Riemannian gradient. Second, define a way to take a step in the direction of the Riemannian gradient while staying *within the manifold*.

The tangent space

Each point W in an embedded manifold $\mathcal{M}$ has a tangent space associated with it, denoted $T_{\mathbf{W}}\mathcal{M}$, as shown in Fig. 1.1 (for a formal definition of the tangent space, see Chapter 2.2, Appendix A). The tangent space is a vector space of the same dimension as the manifold that can be identified

in a natural way with a linear subspace of the ambient space. Let P_W denote the linear projection operator onto the tangent space $T_{\mathbf{W}}\mathcal{M}$.

Given a manifold $\mathcal{M}$ and a differentiable function $\mathcal{L} : \mathcal{M} \to \mathbb{R}$, the *Riemannian gradient* $\nabla\mathcal{L}(W)$ of $\mathcal{L}$ on $\mathcal{M}$ at a point $\mathbf{W}$ is a vector in the tangent space $T_{\mathbf{W}}\mathcal{M}$. A very useful property of embedded manifolds is the following: given a differentiable function f defined on the ambient space (and thus on the manifold), the Riemannian gradient of f at point W is simply the linear projection P_W of the ordinary gradient of f onto the tangent space $T_W\mathcal{M}$.

Thus, if we denote the regular gradient of $\mathcal{L}$ in $\mathbb{R}^{n\times m}$ by $\tilde{\nabla}\mathcal{L}$, we have

$$\nabla\mathcal{L}(W) = P_W(\tilde{\nabla}\mathcal{L}).$$

An important consequence follows in case the manifold represents the set of points obeying a certain constraint. In this case the Riemannian gradient of f is equivalent to the ordinary gradient of f minus the component which is normal to the constraint. Indeed this normal component is exactly the component which is irrelevant when performing constrained optimization. See Figure 1.1.

Consider the $t+1$ step of an iterative gradient update procedure intended to minimize $\mathcal{L}(W)$ over the manifold $\mathcal{M}$. The Riemannian gradient allows us to compute an update $W^{t+\frac{1}{2}} = W^t - \eta\nabla\mathcal{L}(W)$, for a given iterate point W^t and step size η. This however is not yet a full Riemannian gradient step, since $W^{t+\frac{1}{2}}$ is almost certainly outside of the manifold $\mathcal{M}$. Observe for example the unit sphere $\mathcal{S}^{n-1}$. The tangent space at point $x \in \mathcal{S}^{n-1}$ is

$$T_x\mathcal{S}^{n-1} = \{z \in \mathbb{R}^n : x^T z = 0\}.$$

The projection operator onto the tangent space $T_x\mathcal{S}^{n-1}$ is $P_x(y) = y - xx^T y$. Let the Riemannian gradient at a point $x \in \mathcal{S}^{n-1}$ be the vector $P_x(y)$. Then the update vector $x - \eta P_x(y)$ will almost certainly not be a point on the unit sphere. The obvious step in this case will be to project $x - \eta P_x(y)$ onto the unit sphere. We will generalize and formalize this idea in the next subsection, where we examine in general how $W^{t+\frac{1}{2}}$ can be mapped back onto the manifold $\mathcal{M}$.

Geodesics and retractions

The natural generalization of straight lines to the manifold context are *geodesic curves*. A geodesic curve is locally the shortest path between two points on the manifold, or equivalently, a curve with no acceleration tangent to the manifold (Absil et al., 2009). The ideal way to map $W^{t+\frac{1}{2}} = W^t - \eta^t\nabla\mathcal{L}(W)$ back to the manifold would be to follow the geodesic curve originating in W^t and going in the direction of $\nabla\mathcal{L}(W)$. This is called the *exponential mapping* (Do Carmo, 1992, chapter 3), and we apply it to the tangent vector $\nabla\mathcal{L}(W)$. Under mild regularity conditions regarding the loss function $\mathcal{L}$ and the manifold $\mathcal{M}$, applying the exponential mapping to the Riemannian gradient is guaranteed to yield a sequence converging to a local optimum. However, for many manifolds, including the low-rank manifold considered considered in Chapter 2.2 below, calculating the geodesic curve and the resultant exponential mapping is computationally expensive (Vandereycken et al., 2009).

A major insight from the field of Riemannian manifold optimization is that one can use a family of mappings called *retractions* which merely approximate the exponential mapping. Using such retractions maintains the convergence properties obtained with the exponential mapping, but is much cheaper computationally for a suitable choice of retraction. Projecting onto the manifold is in itself a form of retraction (Absil and Malick, 2012), however more efficient or stable retractions can often be devised. Note that projection on the manifold is different, and usually more difficult, than projection onto the tangent space, since the manifold is typically a non-linear structure,

unlike the tangent space. In Chapter 2.2 we introduce a new retraction for the low-rank manifold, and then use it to derive a Riemannian stochastic gradient algorithm. In Chapter 2.4, we show how the exact exponential mapping for the orthogonal matrix manifold can be decomposed into very efficient updates based on Givens rotations (see also Subsection 1.2.3 above). We show these updates to be a Riemannian equivalent of coordinate descent on the orthogonal manifold, and apply them to learn sparse PCA models and to perform robust orthogonal tensor decomposition.

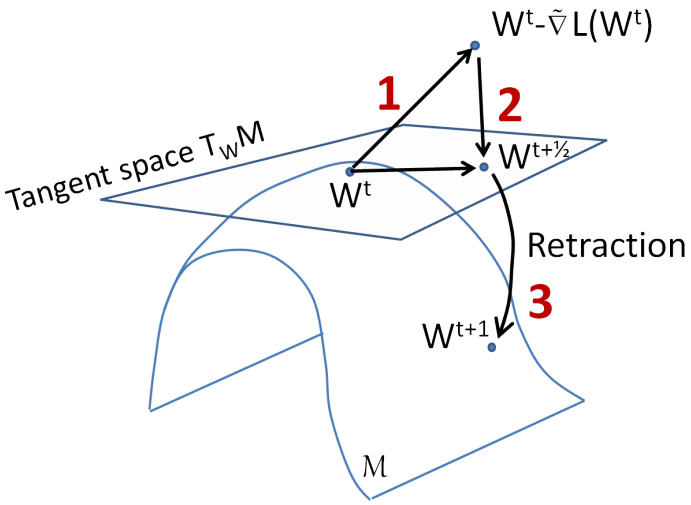


Figure 1.1: A three step procedure for computing a retracted gradient at point W^t. Step 1: ordinary gradient step. Step 2: linearly project ambient gradient onto tangent space $T_{W^t}\mathcal{M}$ in order to get the Riemannian gradient step $W^{t+\frac{1}{2}}$. Step 3: retract the Riemannian gradient step back to the manifold.

Chapter 2

Results

In this Chapter I present the main results of this thesis. The Chapter is comprised of four published papers, as follows:

- Chapter 2.1: Large Scale Online Learning of Image Similarity Through Ranking. Gal Chechik, Varun Sharma*, Uri Shalit* and Samy Bengio (* equal contribution). Journal of Machine Learning, 2010.
- Chapter 2.2: Online Learning in the Embedded Manifold of Low-rank Matrices. Uri Shalit, Daphna Weinshall and Gal Chechik. Journal of Machine Learning, 2012.
- Chapter 2.3: FuncISH: learning a functional representation of neural ISH images. Noa Liscovitch*, Uri Shalit* and Gal Chechik (* equal contribution). Bioinformatics, 2013.
- Chapter 2.4: Coordinate-descent for learning orthogonal matrices through Givens rotations. Uri Shalit and Gal Chechik. International Conference on Machine Learning (ICML), 2014.

2.1 Large Scale Online Learning of Image Similarity Through Ranking

Journal of Machine Learning Research 11 (2010) 1109-1135 Submitted 2/09; Revised 9/09; Published 3/10

Large Scale Online Learning of Image Similarity Through Ranking

Gal Chechik GAL@GOOGLE.COM
Google
1600 Amphitheatre Parkway
Mountain View CA, 94043

Varun Sharma[*] VASHARMA@GOOGLE.COM
Google, RMZ Infinity
Old Madras Road, Bengalooru
Karnataka 560016, India

Uri Shalit[*†] URI.SHALIT@MAIL.HUJI.AC.IL
The Gonda Brain Research Center
Bar Ilan University
52900, Israel

Samy Bengio BENGIO@GOOGLE.COM
Google
1600 Amphitheatre Parkway
Mountain View CA, 94043

Editor: Soeren Sonnenburg, Vojtech Franc, Elad Yom-Tov, Michele Sebag

Abstract

Learning a measure of similarity between pairs of objects is an important generic problem in machine learning. It is particularly useful in large scale applications like searching for an image that is similar to a given image or finding videos that are relevant to a given video. In these tasks, users look for objects that are not only visually similar but also semantically related to a given object. Unfortunately, the approaches that exist today for learning such semantic similarity do not scale to large data sets. This is both because typically their CPU and storage requirements grow quadratically with the sample size, and because many methods impose complex positivity constraints on the space of learned similarity functions.

The current paper presents OASIS, an *Online Algorithm for Scalable Image Similarity* learning that learns a bilinear similarity measure over sparse representations. OASIS is an online dual approach using the passive-aggressive family of learning algorithms with a large margin criterion and an efficient hinge loss cost. Our experiments show that OASIS is both fast and accurate at a wide range of scales: for a data set with thousands of images, it achieves better results than existing state-of-the-art methods, while being an order of magnitude faster. For large, web scale, data sets, OASIS can be trained on more than two million images from 150K text queries within 3 days on a single CPU. On this large scale data set, human evaluations showed that 35% of the ten nearest neighbors of a given test image, as found by OASIS, were semantically relevant to that image. This suggests that query independent similarity could be accurately learned even for large scale data sets that could not be handled before.

Keywords: large scale, metric learning, image similarity, online learning

∗. Varun Sharma and Uri Shalit contributed equally to this work.
†. Also at ICNC, The Hebrew University of Jerusalem, 91904, Israel.

1. Introduction

Large scale learning is sometimes defined as the regime where learning is limited by computational resources rather than by availability of data (Bottou, 2008). Learning a pairwise similarity measure is a particularly challenging large scale task: since pairs of samples have to be considered, the large scale regime is reached even for fairly small data sets, and learning similarity for large data sets becomes exceptionally hard to handle.

At the same time, similarity learning is a well studied problem with multiple real world applications. It is particularly useful for applications that aim to discover new and relevant data for a user. For instance, a user browsing a photo in her album may ask to find similar or related images. Another user may search for additional data while viewing an online video or browsing text documents. In all these applications, similarity could have different flavors: a user may search for images that are similar visually, or semantically, or anywhere in between.

Many similarity learning algorithms assume that the available training data contains real-valued pairwise similarities or distances. However, in all the above examples, the precise numerical value of pairwise similarity between objects is usually not available. Fortunately, one can often obtain information about the *relative* similarity of different pairs (Frome et al., 2007), for instance, by presenting people with several object pairs and asking them to select the pair that is most similar. For large scale data, where man-in-the-loop experiments are prohibitively costly, relative similarities can be extracted from analyzing pairs of images that are returned in response to the same text query (Schultz and Joachims, 2004). For instance, the images that are ranked highly by one of the image search engines for the query "cute kitty" are likely to be semantically more similar than a random pair of images. The current paper focuses on this setting: similarity information is extracted from pairs of images that share a common label or are retrieved in response to a common text query.

Similarity learning has an interesting reciprocal relation with classification. On one hand, pairwise similarity can be used in classification algorithms like nearest neighbors or kernel methods. On the other hand, when objects can be classified into (possibly overlapping) classes, the inferred labels induce a notion of similarity across object pairs. Importantly however, similarity learning assumes a form of supervision that is weaker than in classification, since no labels are provided. OASIS is designed to learn a *class-independent* similarity measure with no need for class labels.

A large number of previous studies have focused on learning a similarity measure that is also a metric, like in the case of a positive semidefinite matrix that defines a Mahalanobis distance (Yang, 2006). However, similarity learning algorithms are often evaluated in a context of ranking. For instance, the learned metric is typically used together with a nearest-neighbor classifier (Weinberger et al., 2006; Globerson and Roweis, 2006). When the amount of training data available is very small, adding positivity constraints for enforcing metric properties is useful for reducing over fitting and improving generalization. However, when sufficient data is available, as in many modern applications, adding positive semi-definitiveness constraints consumes considerable computation time, and its benefit in terms of generalization are limited. With this view, we take here an approach that avoids imposing positivity or symmetry constraints on the learned similarity measure.

The current paper presents an approach for learning semantic similarity that scales up to an order of magnitude larger than current published approaches. Three components are combined to make this approach fast and scalable: First, our approach uses an unconstrained bilinear similarity. Given two images p_1 and p_2 we measure similarity through a bilinear form $p_1^T \mathbf{W} p_2$, where the matrix $\mathbf{W}$ is not required to be positive, or even symmetric. Second we use a sparse representation

of the images, which allows to compute similarities very fast. Finally, the training algorithm that we developed, OASIS, *Online Algorithm for Scalable Image Similarity learning*, is an online dual approach based on the passive-aggressive algorithm (Crammer et al., 2006). It minimizes a large margin target function based on the hinge loss, and already converges to high quality similarity measures after being presented with a small fraction of the training pairs.

We find that OASIS is both fast and accurate at a wide range of scales: for a standard benchmark with thousands of images, it achieves better (but comparable) results than existing state-of-the-art methods, with computation times that are shorter by orders of magnitude. For web-scale data sets, OASIS can be trained on more than two million images within three days on a single CPU, and its training time grows linearly with the size of the data. On this large scale data set, human evaluations of OASIS learned similarity show that 35% of the ten nearest neighbors of a given image are semantically relevant to that image.

The paper is organized as follows. We first present our online algorithm, OASIS, based on the Passive-aggressive family of algorithms. We then present the sparse feature extraction technique used in the experiments. We continue by describing experiments with OASIS on problems of image similarity, at two different scales: a large scale academic benchmark with tens of thousands of images, and a web-scale problem with millions of images. The paper ends with a discussion on properties of OASIS.

2. Learning Relative Similarity

We consider the problem of learning a pairwise similarity function S, given data on the relative similarity of pairs of images.

Formally, let $\mathcal{P}$ be a set of images, and $r_{ij} = r(p_i, p_j) \in \mathbb{R}$ be a pairwise relevance measure which states how strongly $p_j \in \mathcal{P}$ is related to $p_i \in \mathcal{P}$. This relevance measure could encode the fact that two images belong to the same category or were appropriate for the same query. We do not assume that we have full access to all the values of r. Instead, we assume that we can compare some pairwise relevance scores (for instance $r(p_i, p_j)$ and $r(p_i, p_k)$) and decide which pair is more relevant. We also assume that when $r(p_i, p_j)$ is not available, its value is zero (since the vast majority of images are not related to each other). Our goal is to learn a similarity function $S(p_i, p_j)$ that assigns higher similarity scores to pairs of more relevant images,

$$S(p_i, p_i^+) > S(p_i, p_i^-)\,, \quad \forall p_i, p_i^+, p_i^- \in \mathcal{P} \text{ such that } r(p_i, p_i^+) > r(p_i, p_i^-). \tag{1}$$

In this paper we overload notation by using p_i to denote both the image and its representation as a column vector $p_i \in \mathbb{R}^d$. We consider a parametric similarity function that has a bi-linear form,

$$S_{\mathbf{W}}(p_i, p_j) \equiv p_i^T \, \mathbf{W} \, p_j \tag{2}$$

with $\mathbf{W} \in \mathbb{R}^{d \times d}$. Importantly, if the images p_i are represented as sparse vectors, namely, only a number $k_i \ll d$ of the d entries in the vector p_i are non-zeroes, then the value of Equation (2) can be computed very efficiently even when d is large. Specifically, $S_{\mathbf{W}}$ can be computed with complexity of $O(k_i k_j)$ regardless of the dimensionality d.

2.1 An Online Algorithm

We propose an online algorithm based on the Passive-Aggressive (PA) family of learning algorithms introduced by Crammer et al. (2006). Here we consider an algorithm that uses triplets of images $p_i, p_i^+, p_i^- \in \mathcal{P}$ such that $r(p_i, p_i^+) > r(p_i, p_i^-)$.

We aim to find a parametric similarity function S such that all triplets obey

$$S_{\mathbf{W}}(p_i, p_i^+) > S_{\mathbf{W}}(p_i, p_i^-) + 1 \tag{3}$$

which means that it fulfills Equation (1) with a safety margin of 1. We define the following hinge loss function for the triplet:

$$l_{\mathbf{W}}(p_i, p_i^+, p_i^-) = \max\left\{0, 1 - S_{\mathbf{W}}(p_i, p_i^+) + S_{\mathbf{W}}(p_i, p_i^-)\right\}. \tag{4}$$

Our goal is to minimize a global loss $L_{\mathbf{W}}$ that accumulates hinge losses (4) over all possible triplets in the training set:

$$L_{\mathbf{W}} = \sum_{(p_i, p_i^+, p_i^-) \in \mathcal{P}} l_{\mathbf{W}}(p_i, p_i^+, p_i^-)\ .$$

In order to minimize this loss, we apply the Passive-Aggressive algorithm iteratively over triplets to optimize $\mathbf{W}$. First, $\mathbf{W}$ is initialized to some value $\mathbf{W}^0$. Then, at each training iteration i, we randomly select a triplet (p_i, p_i^+, p_i^-), and solve the following convex problem with soft margin:

$$\begin{aligned} \mathbf{W}^i &= \operatorname*{argmin}_{\mathbf{W}} \frac{1}{2}\|\mathbf{W} - \mathbf{W}^{i-1}\|_{Fro}^2 + C\xi \\ \text{s.t.} \quad & l_{\mathbf{W}}(p_i, p_i^+, p_i^-) \le \xi \quad \text{and} \quad \xi \ge 0 \end{aligned} \tag{5}$$

where $\|\cdot\|_{Fro}$ is the Frobenius norm (point-wise L_2 norm). Therefore, at each iteration i, $\mathbf{W}^i$ is selected to optimize a trade-off between remaining close to the previous parameters $\mathbf{W}^{i-1}$ and minimizing the loss on the current triplet $l_{\mathbf{W}}(p_i, p_i^+, p_i^-)$. The *aggressiveness* parameter C controls this trade-off.

OASIS

Initialization:

Initialize $\mathbf{W}^0 = I$

Iterations

repeat

Sample three images p, p_i^+, p_i^-, such that $r(p_i, p_i^+) > r(p_i, p_i^-)$.

Update $\mathbf{W}^i = \mathbf{W}^{i-1} + \tau_i \mathbf{V}^i$

where $\tau_i = \min\left\{C, \frac{l_{\mathbf{W}^{i-1}}(p_i, p_i^+, p_i^-)}{\|\mathbf{V}^i\|^2}\right\}$

and $\mathbf{V}^i = [p_i^1(p_k^+ - p_k^-), \ldots, p_i^d(p_k^+ - p_k^-)]^T$

until (stopping criterion)

Figure 1: Pseudo-code of the OASIS algorithm.

We follow Crammer et al. (2006) to solve the problem in Equation (5). When $l_{\mathbf{W}}(p_i, p_i^+, p_i^-) = 0$, it is clear that $\mathbf{W}^i = \mathbf{W}^{i-1}$ satisfies Equation (5) directly. Otherwise, we define the Lagrangian

$$\mathcal{L}(\mathbf{W}, \tau, \xi, \lambda) = \frac{1}{2}\|\mathbf{W} - \mathbf{W}^{i-1}\|^2 + C\xi + \tau(1 - \xi - p_i^T \mathbf{W}(p_i^+ - p_i^-)) - \lambda\xi \tag{6}$$

where $\tau \geq 0$ and $\lambda \geq 0$ are Lagrange multipliers. The optimal solution is such that the gradient vanishes $\frac{\partial \mathcal{L}(\mathbf{W}, \tau, \xi, \lambda)}{\partial \mathbf{W}} = 0$, hence

$$\frac{\partial \mathcal{L}(\mathbf{W}, \tau, \xi, \lambda)}{\partial \mathbf{W}} = \mathbf{W} - \mathbf{W}^{i-1} - \tau \mathbf{V}_i = 0$$

where the gradient matrix $\mathbf{V}_i = \frac{\partial \mathcal{L}_W}{\partial \mathbf{W}} = [p_i^1(p_i^+ - p_i^-), \ldots, p_i^d(p_i^+ - p_i^-)]^T$. The optimal new $\mathbf{W}$ is therefore

$$\mathbf{W} = \mathbf{W}^{i-1} + \tau \mathbf{V}_i \tag{7}$$

where we still need to estimate τ. Differentiating the Lagrangian with respect to ξ and setting it to zero also yields:

$$\frac{\partial \mathcal{L}(\mathbf{W}, \tau, \xi, \lambda)}{\partial \xi} = C - \tau - \lambda = 0 \tag{8}$$

which, knowing that $\lambda \geq 0$, means that $\tau \leq C$. Plugging Equations (7) and (8) back into the Lagrangian in Equation (6), we obtain

$$\mathcal{L}(\tau) = \frac{1}{2}\tau^2 \|\mathbf{V}_i\|^2 + \tau(1 - p_i^T(\mathbf{W}^{i-1} + \tau \mathbf{V}_i)(p_i^+ - p_i^-)) \,.$$

Regrouping the terms we obtain

$$\mathcal{L}(\tau) = -\frac{1}{2}\tau^2 \|\mathbf{V}_i\|^2 + \tau(1 - p_i^T \mathbf{W}^{i-1}(p_i^+ - p_i^-)) \,.$$

Taking the derivative of this second Lagrangian with respect to τ and setting it to 0, we have

$$\frac{\partial \mathcal{L}(\tau)}{\partial \tau} = -\tau \|\mathbf{V}_i\|^2 + (1 - p_i^T \mathbf{W}^{i-1}(p_i^+ - p_i^-)) = 0$$

which yields

$$\tau = \frac{1 - p_i^T \mathbf{W}^{i-1}(p_i^+ - p_i^-)}{\|\mathbf{V}_i\|^2} = \frac{l_{\mathbf{W}^{i-1}}(p_i, p_i^+, p_i^-)}{\|\mathbf{V}_i\|^2} \,.$$

Finally, Since $\tau \leq C$, we obtain

$$\tau = \min\left\{C, \frac{l_{\mathbf{W}^{i-1}}(p_i, p_i^+, p_i^-)}{\|\mathbf{V}_i\|^2}\right\} \,. \tag{9}$$

Equations (7) and (9) summarize the update needed for every triplets (p_i, p_i^+, p_i^-). It has been shown (Crammer et al., 2006) that applying such an iterative algorithm yields a cumulative online loss that is likely to be small. It was furthermore shown that selecting the best $\mathbf{W}_i$ during training using a hold-out validation set achieves good generalization. We also show below that multiple runs of the algorithm converge to provide similar precision (see Figure 7).

2.2 Loss Bounds

Following closely the analysis of loss bounds for passive aggressive (PA) algorithms developed by Crammer et al. (2006) we state similar *relative* bounds for the OASIS framework. We do this by rewriting OASIS as a straightforward linear classification problem. Denote by $\overrightarrow{w_i}$ the vector obtained by"unfolding" the matrix $\mathbf{W}$ (concatenating all its columns into a single vector) and similarly $\overrightarrow{x_i}$ the unfolded matrix $p_i(p_i^+ - p_i^-)^T$. Using this notation, the constraint in Equation (3) becomes

$$\overrightarrow{w_i} \cdot \overrightarrow{x_i} > 1 \quad ,$$

with $\cdot$ denoting the standard inner product. This is equivalent to the formulation of PA when the label y_i is always 1. The introduction of slack variables in Equation (5) brings us to the variant denoted by Crammer et al. (2006) as PA-I.

The loss bounds in Crammer et al. (2006) rely on $\overrightarrow{w_0}$ being the zero vector. Since here we initialize with $W^0 = I$ (the identity matrix) we need to adapt the analysis slightly. Let $\overrightarrow{u}$ be a vector in $\mathbb{R}^{d^2}$ obtained by unfolding an arbitrary matrix $\mathbf{U}$. We define

$$l_i = 1 - \overrightarrow{w_i} \cdot \overrightarrow{x_i} \quad \text{and} \quad l_i^* = 1 - \overrightarrow{u} \cdot \overrightarrow{x_i} \quad ,$$

where l_i is the instantaneous loss at round i, and l_i^* is the loss suffered by the arbitrary vector $\overrightarrow{u}$. The following two theorems rely on Lemma 1 of Crammer et al. (2006), which we restate without proof:

$$\sum \tau_i (2l_i - \tau_i \|x_i\|^2 - 2l_i^*) \leq \|\overrightarrow{u} - \overrightarrow{w_0}\|^2 \, .$$

While in Crammer et al. (2006) $\overrightarrow{w_0}$ is the zero vector, in our case $\overrightarrow{w_0}$ is the unfolded *identity matrix*. We therefore have

$$\|\overrightarrow{u} - \overrightarrow{w_0}\|^2 = \|\mathbf{U}\|_{Fro}^2 - 2trace(\mathbf{U}) + n \, .$$

Using this modified lemma we can restate the relevant bound:

Theorem 1 *Let $(\overrightarrow{x_1}),...,(\overrightarrow{x_M})$ be a sequence of examples where $\overrightarrow{x_i} \in \mathbb{R}^{d^2}$, $\|\overrightarrow{x_i}\| \leq R$ for all $i = 1...M$. Then, for any matrix $\mathbf{U} \in \mathbb{R}^{n^2}$, the number of prediction mistakes made by OASIS on this sequence of examples is bounded from above by,*

$$\max\{R^2, 1/C\} \left(\|\mathbf{U}\|_{Fro}^2 - 2trace(\mathbf{U}) + n + 2C \sum_{i=1}^{M} l_i^* \right)$$

where C is the aggressiveness parameter provided to OASIS.

2.3 Sampling Strategy

For real world data sets, the actual number of triplets (p_i, p_i^+, p_i^-) is typically very large and cannot be stored in memory. Instead, we use the fact that the number of relevant images for a category or a query is typically small, and keep a list of relevant images for each query or category. For the case of single-labeled images, we can efficiently retrieve an image that is relevant to a given image, by first finding its class, and then finding another image from that class. The case of multi-labeled images is described in Section 5.2.

Specifically, to sample a triplet (p_i, p_i^+, p_i^-) during training, we first uniformly sample an image p_i from $\mathcal{P}$. Then we uniformly sample an image p_i^+ from the images sharing the same categories

or queries as p_i. Finally, we uniformly sample an image p_i^- from the images that share no category or query with p_i. When the set $\mathcal{P}$ is very large and the number of categories or queries is also very large, one does not need to maintain the set of non-relevant images for each image: sampling directly from $\mathcal{P}$ instead only adds a small amount of noise to the training procedure and is not really harmful.

When relevance feedbacks $r(p_i, p_j)$ are provided as real numbers and not just $\in \{0,1\}$, one could use these number to bias training towards those pairs that have a higher relevance feedback value. This can be done by considering $r(p_i, p_j)$ as frequencies of appearance, and sampling pairs according to the distribution of these frequencies.

3. Image Representation

The problem of selecting an informative representation of images is still an unsolved computer vision challenge, and an ongoing research topic. Different approaches for image representation have been proposed including by Feng et al. (2004); Takala et al. (2005) and Tieu and Viola (2004). In the information retrieval community there is wide agreement that a bag-of-words representation is a very useful representation for handling text documents in a wide range of applications. For image representation, there is still no such approach that would be adequate for a wide variety of image processing problems. However, among the proposed representations, a consensus is emerging on using *local descriptors* for various tasks, for example, Lowe (2004); Quelhas et al. (2005). This type of representation segments the image into *regions of interest*, and extracts visual features from each region. The segmentation algorithm as well as the region features vary among approaches, but, in all cases, the image is then represented as a set of feature vectors describing the regions of interest. Such a set is often called a *bag-of-local-descriptors*.

In this paper we take the approach of creating a sparse representation based on the framework of local descriptors. Our features are extracted by dividing each image into overlapping square blocks, and each block is then described with edge and color histograms. For edge histograms, we rely on *uniform Local Binary Patterns* (uLBPs) proposed by Ojala et al. (2002). These texture descriptors have shown to be effective on various tasks in the computer vision literature (Ojala et al., 2002; Takala et al., 2005), certainly due to their robustness with respect to changes in illumination and other photometric transformations (Ojala et al., 2002). Local Binary Patterns estimate a texture histogram of a block by considering differences in intensity at circular neighborhoods centered on each pixel. Precisely, we use $LBP_{8,2}$ patterns, which means that a circle of radius 2 is considered centered on each block. For each circle, the intensity of the center pixel is compared to the interpolated intensities located at 8 equally-spaced locations on the circle, as shown on Figure 2, left. These eight binary tests (lower or greater intensity) result in an 8-bit sequence, see Figure 2, right. Hence, each block pixel is mapped to a sequence among $2^8 = 256$ possible sequences and each block can therefore be represented as a 256-bin histogram. In fact, it has been observed that the bins corresponding to non-uniform sequences (sequences with more than 2 transitions $1 \to 0$ or $0 \to 1$) can be merged, yielding more compact 59-bin histograms without performance loss (Ojala et al., 2002).

Color histograms are obtained by K-means clustering. We first select a palette or typical colors by training a color codebook from the Red-Green-Blue pixels of a large training set of images using K-means. The color histogram of a block is then obtained by mapping each block pixel to the closest color in the codebook palette.

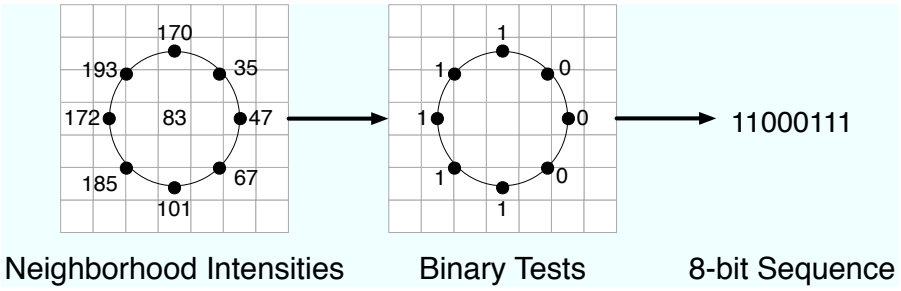


Figure 2: An example of Local Binary Pattern ($LBP_{8,2}$). For a given pixel, the Local Binary Pattern is an 8-bit code obtained by verifying whether the intensity of the pixel is greater or lower than its 8 neighbors.

Finally, the histograms describing color and edge statistics of each block are concatenated, which yields a single vector descriptor per block. Our local descriptor representation is therefore simple, relying on both a basic segmentation approach and simple features. Naturally, alternative representations could also be used with OASIS, (Feng et al., 2004; Grangier et al., 2006; Tieu and Viola, 2004) However, this paper focuses on the learning model, and a benchmark of image representations is beyond the scope of the current paper.

As a final step, we use the representation of blocks to obtain a representation for an image. For computation efficiency we aim at a high dimensional and sparse vector space. For this purpose, each local descriptor of an image p is represented as a discrete index, called *visual term* or *visterm*, and, like for text data, the image is represented as a *bag-of-visterms* vector, in which each component p_i is related to the presence or absence of visterm i in p.

The mapping of the descriptors to discrete indexes is performed according to a codebook C, which is typically learned from the local descriptors of the training images through k-means clustering (Duygulu et al., 2002; Jeon and Manmatha, 2004; Quelhas et al., 2005). The assignment of the weight p_i of visterm i in image p is as follows:

$$p_i = \frac{f_i \, d_i}{\sqrt{\sum_{j=1}^{d} (f_j \, d_j)^2}} \, ,$$

where f_i is the term frequency of i in p, which refers to the number of occurrences of i in p, while d_j is the inverse document frequency of j, which is defined as $-log(r_j)$, r_j being the fraction of training images containing at least one occurrence of visterm j. This approach has been found successful for the task of content based image ranking described by Grangier and Bengio (2008).

In the experiments described below, we used a large set of images collected from the web to train the features. This set is described in more detail in Section 5.2. We used a set of 20 typical RGB colors (hence the number of clusters used in the k-means for colors was 20), the block vocabulary size $d = 10000$ and our image blocks were of size 64x64 pixels, overlapping every 32 pixels. Furthermore, in order to be robust to scale, we extracted blocks at various scales by successively down scaling images by a factor of 1.25 and extracting the features at each level, until there were less than 10 blocks in the resulting image. There was on average around 70 non-zero

values (out of 10000) describing a single image. Note that no other information (such as meta-data) was added in the input vector representation each image.

4. Related Work

Similarity learning can be considered in two main setups, depending on the type of available training labels. First, a regression setup, where the training set consists of pairs of objects x_i^1, x_i^2 and their pairwise similarity $y_i \in \mathbf{R}$. In many cases however, precise similarities are not available, but rather a weaker notion of similarity order. In one such setup, the training set consists of triplets of objects x_i^1, x_i^2, x_i^3 and a ranking similarity function, that can tell which of the two pairs (x^1, x^2) or (x^1, x^3) is more similar. Finally, multiple similarity learning studies assume that a binary measure of similarity is available $y_i \in \{+1, -1\}$, indicating whether a pair of objects is similar or not.

For small-scale data, there are two main groups of similarity learning approaches. The first approach, learning Mahalanobis distances, can be viewed as learning a linear projection of the data into another space (often of lower dimensionality), where a Euclidean distance is defined among pairs of objects. Such approaches include Fisher's Linear Discriminant Analysis (LDA), relevant component analysis (RCA) (Bar-Hillel et al., 2003), supervised global metric learning (Xing et al., 2003), large margin nearest neighbor (LMNN) (Weinberger et al., 2006) and Metric Learning by Collapsing Classes (Globerson and Roweis, 2006). A Mahalanobis distance learning algorithm which uses a supervision signal identical to the one we employ in OASIS is Rosales and Fung (2006), which learns a special kind of PSD matrix via linear programming. See also a review by Yang (2006) for more details.

The second family of approaches, learning kernels, is used to improve performance of kernel based classifiers. Learning a full kernel matrix in a non parametric way is prohibitive except for very small data. As an alternative, several studies suggested to learn a weighted sum of pre-defined kernels (Lanckriet et al., 2004) where the weights are being learned from data. In some applications this was shown to be inferior to uniform weighting of the kernels (Noble, 2008). The work of Frome et al. (2007) further learns a weighting over local distance function for every image in the training set. Non linear image similarity learning was also studied in the context of dimensionality reduction, as in Hadsell et al. (2006).

Finally, Jain et al. (2008a,b), based on work by Davis et al. (2007), aim to learn metrics in an online setting. This work is one of the closest work with respect to OASIS: it learns a linear model of a [dis-]similarity function between documents in an online way. The main difference is that the work of Jain et al. (2008a) learn a true distance throughout the learning process, imposing positive definiteness constraints, and is slightly less efficient computationally. We argue in this paper that in the large scale regime, such a constraint is not necessary given the amount of available training examples.

Another work closely related to OASIS is that of Rasiwasia and Vasconcelos (2008), which also tries to learn a semantic similarity function between images. In their case, however, semantic similarity is learned by representing each image by the posterior probability distribution over a pre-defined set of semantic tags, and then computing the distance between two images as the distance between the two underlying posterior distributions. The representation size of images in this approach is therefore equal to the number of semantic classes, hence it will not scale when the number of semantic classes is very large as in free text search.

5. Experiments

Evaluating large scale learning algorithms poses special challenges. First, current available benchmarks are limited either in their scale, like 30K images in Caltech256 as described by Griffin et al. (2007), or in their resolution, such as the tiny images data set of Torralba et al. (2007). Large scale methods are not expected to perform particularly well on small data sets, since they are designed to extract limited information from each sample. Second, many images on the web cannot be used without explicit permission, hence they cannot be collected and packed into a single database. Large, proprietary collections of images do exist, but are not available freely for academic research. Finally, except for very few cases, similarity learning approaches in current literature do not scale to handle large data sets effectively, which makes it hard to compare a new large scale method with the existing methods.

To address these issues, this paper takes the approach of conducting experiments at two different scales. First, to demonstrate the scalability of OASIS we applied OASIS to a web-scale data with 2.7 million images. Second, to investigate the properties of OASIS more deeply, we compare OASIS with small-scale methods using the standard Caltech256 benchmark.

5.1 Evaluation Measures

We evaluated the performance of all algorithms using standard ranking precision measures based on nearest neighbors. For each query image in the test set, all other test images were ranked according to their similarity to the query image. The number of same-class images among the top k images (the k nearest neighbors) was computed. When averaged across test images (either within or across classes), this yields a measure known as precision-at-top-k, providing a precision curve as a function of the rank k.

We also calculated the *mean average precision* (mAP), a measure that is widely used in the information retrieval community. To compute average precision, the precision-at-top-k is first calculated for each test image. Then, it is averaged over all positions k that have a positive sample. For example, if all positives are ranked highest, the average-precision is 1. The average-precision measure is then further averaged across all test image queries, yielding the *mean average precision* (mAP).

5.2 Web-Scale Experiment

Our first set of experiments is based on Google proprietary data that is two orders of magnitude larger than current standard benchmarks. We collected a set of $\sim$150K text queries submitted to the Google Image Search system. For each of these queries, we had access to a set of relevant images, each of which is associated with a numerical relevance score. This yielded a total of $\sim$2.7 million images, which we split into a training set of 2.3 million images and a test set of 0.4 million images (see Table 1).

Set	Number of Queries	Number of Images
Training	139944	2292259
Test	41877	402164

Table 1: Statistics of the Web data set.

5.2.1 EXPERIMENTAL SETUP

We used the query-image relevance information to create an image-image relevance as follows. Denote the set of text queries by $\mathcal{Q}$ and the set of images by $\mathcal{P}$. For each $q \in \mathcal{Q}$, let $\mathcal{P}_q^+$ denote the set of images that are relevant to the query q, and let $\mathcal{P}_q^-$ denote the set of irrelevant images. The query-image relevance is defined by the matrix $\mathbf{R}_{QI} : \mathcal{Q} \times \mathcal{P} \to \mathbb{R}^+$, and obeys $\mathbf{R}_{QI}(q, p_q^+) > 0$ and $\mathbf{R}_{QI}(q, p_q^-) = 0$ for all $q \in \mathcal{Q}$, $p_q^+ \in \mathcal{P}_q^+$, $p_q^- \in \mathcal{P}_q^-$. We also computed a normalized version of $\mathbf{R}_{QI}$, which can be interpreted as a joint distribution matrix, or the probability to observe a query q and an image p for that query,

$$Pr(q,p) = \frac{\mathbf{R}_{QI}(q,p)}{\sum_{q',p'} \mathbf{R}_{QI}(q',p')} \quad .$$

In order to compute the image-image relevance matrix $\mathbf{R}_{II} : \mathcal{P} \times \mathcal{P} \to \mathbb{R}^+$, we treated images as being conditionally independent given the queries, $Pr(p_1, p_2|q) = Pr(p_1|q)Pr(p_2|q)$, and computed the joint image-image probability as a relevance measure

$$Pr(p_1,p_2) = \sum_{q \in \mathcal{Q}} Pr(p_1,p_2|q)\,Pr(q) = \sum_{q \in \mathcal{Q}} Pr(p_1 \mid q)Pr(p_2 \mid q)Pr(q) \; .$$

To improve scalability, we used a threshold over this joint distribution, and considered two images to be related only if their joint distribution exceeded a cutoff value θ

$$\mathbf{R}_{II}(p_1,p_2) = [Pr(p_1,p_2)]_\theta \tag{10}$$

where $[x]_\theta = x$ for $x > \theta$ and is zero otherwise. To set the value of θ we have manually inspected a small subset of pairs of related images taken from the training set. We selected the largest θ such that most of those related pairs had scores above the threshold, while minimizing noise in $\mathbf{R}_{II}$.

Equation 10 is written as if one needs to calculate the full joint matrix $\mathbf{R}_{II}$, but this matrix grows quadratically with the number of images. In practice, we can use the fact that $\mathbf{R}_{QI}$ is very sparse, to quickly create a list with images that are relevant to a given image. To do this given an image p_i, we go over all the queries for which it is relevant $\mathbf{R}_{QI}(q, p_i)$, and for each of these queries, collect the list of all images that are relevant to that query. The average number of queries relevant for an image in our data is small (about 100), and so is the number of images relevant for a given query. As a result, $\mathbf{R}_{II}$ can be calculated efficiently even for large image sets.

We trained OASIS over 2.3 million images in the training set using the sampling mechanism based on the relevance of each image, as described in Section 2.3. To select the number of training iterations, we used as a validation set a small subset of the training set to trace the mean average precision of the model at regular intervals during the training process. Training was stopped when the mean average precision had saturated, which happened after 160 million iterations (triplets). Overall, training took a total of $\sim$4000 minutes on a single CPU of a standard modern machine. Finally, we evaluated the trained model on the 400 thousand images of the test set.

5.2.2 RESULTS

We start with specific examples illustrating the behavior of OASIS, and continue with a quantitative analysis of precision and speed. Table 2 shows the top five images as ranked by OASIS on four examples of query-images in the test set. The relevant text queries for each image are shown beneath the image. The first example (top row), shows a query-image that was originally retrieved

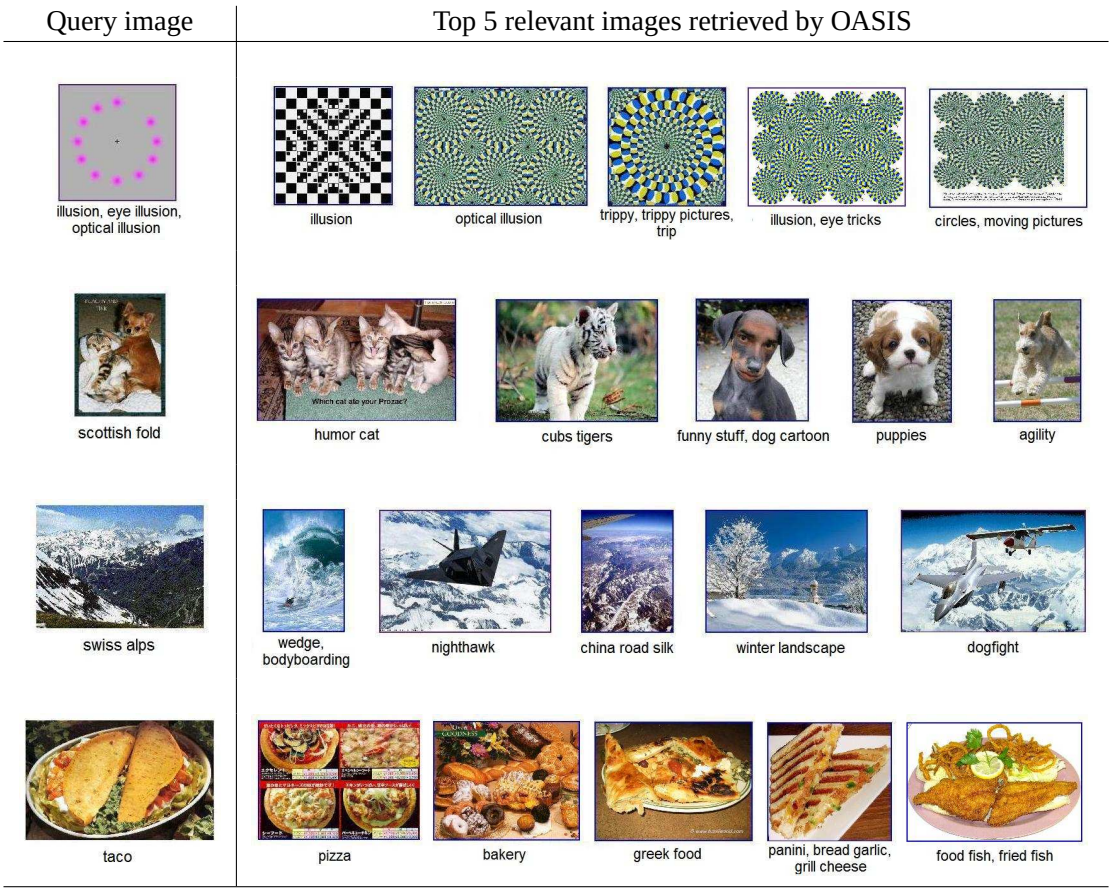


Table 2: OASIS: Successful cases from the Web data set

in response to the text query "illusion". All five images ranked highly by OASIS are semantically related, showing other types of visual illusions. Similar results can be observed for the three remaining examples on this table, where OASIS captures well the semantics of animal photos (cats and dogs), mountains and different food items.

In all these cases, OASIS captures similarity that is both semantic and visual, since the raw visual similarity of these images is not high. A different behavior is demonstrated in Table 3. It shows three cases where OASIS was biased by visual similarity and provided high rankings to images that were semantically non relevant. In the first example, the assortment of flowers is confused with assortments of food items and a thigh section (5th nearest neighbor) which has visually similar shape. The second example presents a query image which in itself has no definite semantic element. The results retrieved are those that merely match texture of the query image and bear no semantic similarity. In the third example, OASIS fails to capture the butterfly in the query image.

To obtain a quantitative evaluation of OASIS we computed the precision at top k, using a threshold $\theta = 0$, which means that an image in the test set is considered relevant to a query image, if there exists at least one text query to which they were both relevant to.

The obtained precision values were quite low, achieving 1.5% precision at the top ranked image. This is drastically lower than the precision described below for Caltech256, and could be the result

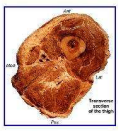
Query image	Top 5 relevant images retrieved by OASIS
 roses bouquet	 dessert  bakery  panini, bread garlic, grill cheese  newt  thigh, muscle group
 garden vegetable	 schwitters  canyon  botswana  canyon grand  know
 insect	 flowers  strawberry  food japanese  chinese food  vegetable fruit, vitamin

Table 3: OASIS: Failure cases from the Web data set

of multiple reasons. First, the number of unique textual queries in our data is very large (around 150K), hence the images in this data set were significantly more heterogeneous than images in the Caltech256 data.

Second, and most importantly, our labels that measure pairwise relevance are very partial. This means that many pairs of images that are semantically related are not labeled as such. A clear demonstration of this effect is observed in Tables 2 and 3. The query images (like "*scottish fold*") have labels that are usually very different from the labels of the retrieved images (as in "*humor cat*", "*agility*") even if their semantic content is very similar. This is a common problem in content-based analysis, since similar content can be described in many different ways. In the case discussed here, the partial data on the query-image relevance $\mathbf{R}_{QI}$ is further propagated to the image-image relevance measure $\mathbf{R}_{II}$.

5.2.3 HUMAN EVALUATION EXPERIMENTS

In order to obtain a more accurate estimate of the real semantic precision, we performed a rating experiment with human evaluators. We chose the 25 most relevant images[1] from the test set and retrieved their 10 nearest neighbors as determined by OASIS. We excluded query-images which contained porn, racy or duplicates in their 10 nearest neighbors. We also selected randomly a set of 10 negative images p^- that were chosen for each of the query images p such that $\mathbf{R}_{II}(p, p^-) = 0$. These negatives were then randomly mixed with the 10 nearest neighbors.

All 25 query images were presented to twenty human evaluators, asking them to mark which of the 20 candidate images are *semantically relevant* to the query image.[2] Evaluators were volunteers

1. The overall relevance of an image was estimated as the sum of relevances of the image with respect to all queries.
2. The description of the task as given to the evaluators is provided in Appendix A.

selected from a pool of friends and colleagues, many of which had experience with search or machine vision problems. We collected the ratings on the positive images and calculated the precision at top k.

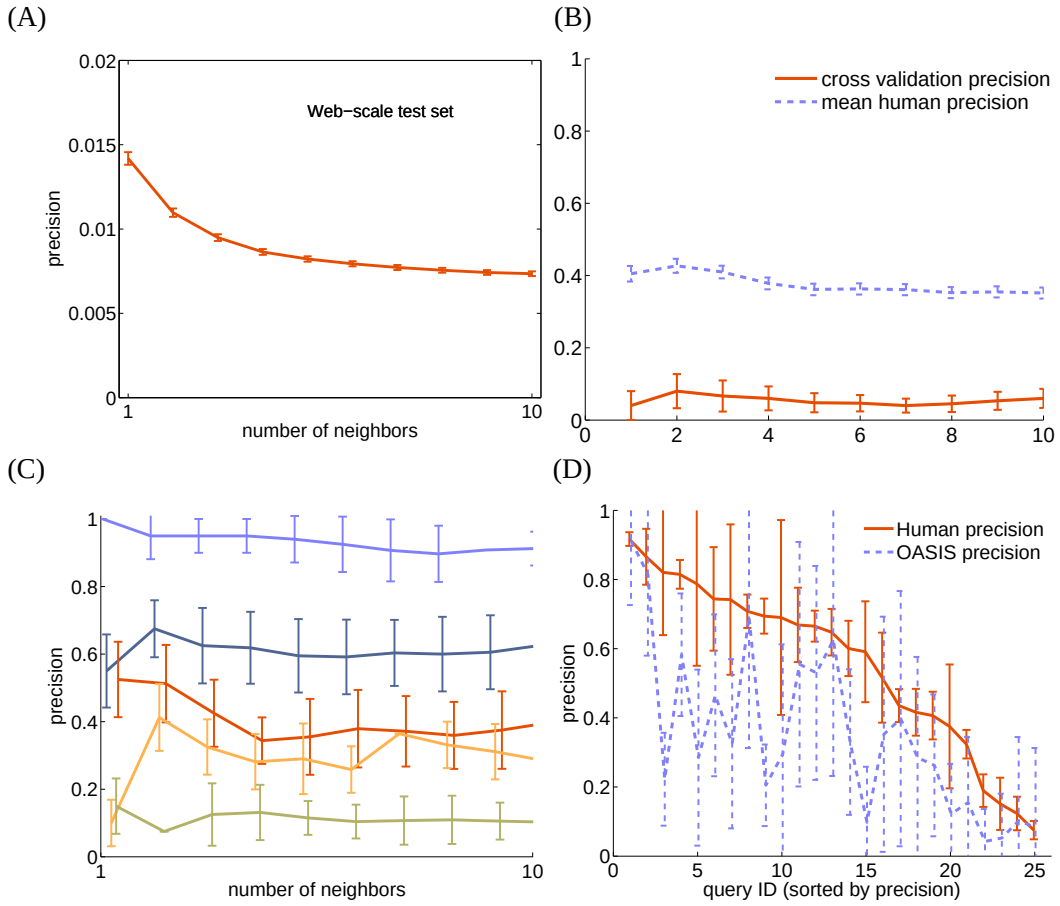


Figure 3: **(A)** Precision at top k as a function of k neighbors computed against $\mathbf{R}_{II}$ ($\theta = 0$) for the web-scale test set. **(B)** Precision at top k as a function of k neighbors for the human evaluation subset. **(C)** Mean precision for 5 selected queries. Error bars denote the standard error of the mean. To select the queries for this plot, we first calculated the mean-average precision per query, sorted the queries by their mAP, and selected the queries ranked at position 1, 6, 11, 16, and 21. **(D)** Precision of OASIS and human evaluators, per query, using rankings of all (remaining) human evaluators as a ground truth.

Figure 3(B) shows the average precision across all queries and evaluators. Precision peaks at 42% and reaches 35% at the top 10 ranked image, being significantly higher than the values calculated automatically using $\mathbf{R}_{II}$.

We observed that the variability across different query images was also very high. Figure 3(C) shows the precision for 5 different queries, selected to span the range of average-precision values. The error bars at each curve show the variability in the responses of different evaluators. The

precision of OASIS varies greatly across different queries. Some query images were "easy" for OASIS, yielding high scores from most evaluators, while other queries retrieved images that were consistently found to be irrelevant by most evaluators.

We also compared the magnitude of variability across human evaluators, with variability across queries. We first calculated the mAP from the precision curves of every query and evaluator, and then calculated the standard deviation in the mAP of every evaluator and of every query. The mean standard deviation over queries was 0.33, suggesting a large variability in the difficulty of image queries, as observed in Figure 3(C) . The mean standard deviation over evaluators was 0.25, suggesting that different evaluators had very different notions of what images should be regarded as "semantically similar" to a query image.

Finally, to estimate an "upper bound" on the difficulty of the task, we also computed the precision of the human evaluators themselves. For every evaluator, we used the rankings of all other evaluators as ground truth, to compute his precision. As with the ranks of OASIS, we computed the fraction of evaluators that marked an image as relevant, and repeated this separately for every query and human evaluator, providing a measure of "coherence" per query. Figure 3(D) shows the mean precision obtained by OASIS and human evaluators for every query in our data. For some queries OASIS achieves precision that is very close to that of the mean human evaluator. In many cases OASIS achieves precision that is as good or better than some evaluators.

5.2.4 SPEED AND SCALABILITY

We further studied how the runtime of OASIS scales with the size of the training set. Figure 4 shows that the runtime of OASIS, as found by early stopping on a separate validation set, grows linearly with the train set size. We compare this to the fastest result we found in the literature, based on a fast implementation of LMNN by Weinberger and Saul (2008). LMNN learns a Mahalanobis distance for k-nearest neighbor classification, aiming to have the nearest neighbors of a sample belong to the same class, and samples from different classes separated by a large margin. The LMNN algorithm is known to scale quadratically with the number of objects, although their experiments with MNIST data show that the active set of constraints grows linearly. This could be because MNIST has 10 classes only. In many real world data however, the number of classes typically grows almost linearly with the number of samples.

5.3 Caltech256 Data Set

To compare OASIS with small-scale methods we used the *Caltech256* data set (Griffin et al., 2007). This data set consists of 30607 images that were obtained from Google image search and from *PicSearch.com*. Images were assigned to 257 categories and evaluated by humans in order to ensure image quality and relevance. After we have pre-processed the images as described in Section 3 and filtered images that were too small, we were left with 29461 images in 256 categories. To allow comparisons with other methods in the literature that were not optimized for sparse representation, we also reduced the block vocabulary size d from 10000 to 1000. This processed data is available online at *http://ai.stanford.edu/∼gal/Research/OASIS*.

Using the Caltech256 data set allows us to compare OASIS with existing similarity learning methods. For OASIS, we treated images that have the same labels as similar. The same labels were used for comparing with methods that learn a metric for classification, as described below.

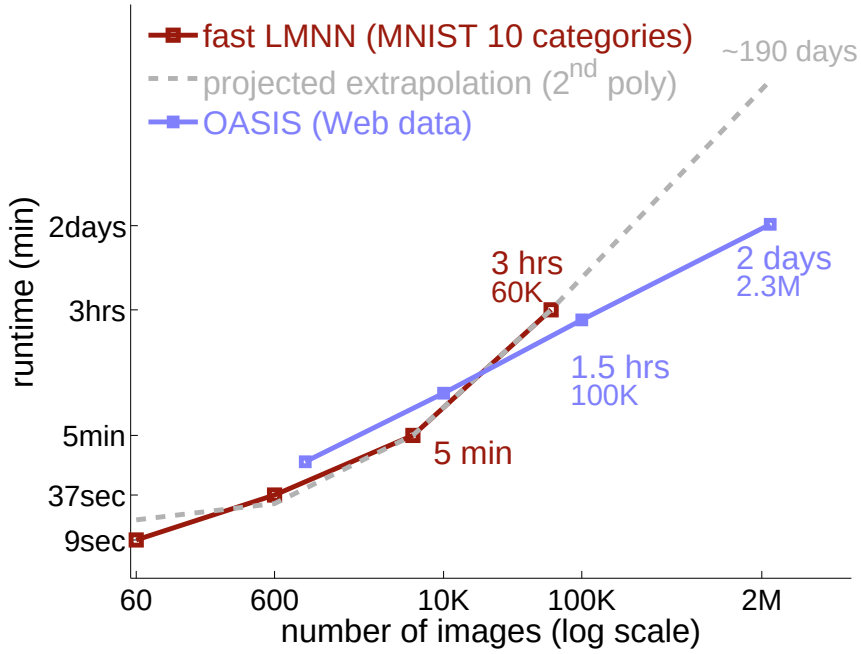


Figure 4: Comparison of the runtime of OASIS and fast-LMNN by Weinberger and Saul (2008), over a wide range of scales. LMNN results (on MNIST data) are faster than OASIS results on subsets of the web data. However LMNN scales quadratically with the number of samples, hence is three times slower on 60K images, and may be infeasible for handling 2.3 million images.

5.3.1 COMPARED METHODS

We compared the following approaches:

1. **OASIS.** - The algorithm described above in Section 2.1.

2. **Euclidean.** - The standard Euclidean distance in feature space. The initialization of OASIS using the identity matrix is equivalent to this distance measure.

3. **MCML** - Metric Learning by Collapsing Classes (Globerson and Roweis, 2006). This approach learns a Mahalanobis distance such that samples from the same class are mapped to the same point. The problem is written as a convex optimization problem, and we have used the gradient-descent implementation provided by the authors.

4. **LMNN** - Large Margin Nearest Neighbor Classification (Weinberger et al., 2006). This approach learns a Mahalanobis distance for k-nearest neighbor classification, aiming to have the k-nearest neighbors of a given sample belong to the same class while examples from different classes are separated by a large margin. As a preprocessing phase, images were projected to a basis of the principal components (PCA) of the data, with no dimensionality reduction, since

this improved the precision results. We also compared with a fast implementation of LMNN, that uses a clever scheme of maintaining a set of active constraints (Weinberger and Saul, 2008). We used the web data discussed above to compare with previously published results obtained with fast-LMNN on MNIST data (see Figure 4).

5. **LEGO** - Online metric learning (Jain et al., 2008a). LEGO learns a Mahalanobis distance in an online fashion using a regularized per instance loss, yielding a positive semidefinite matrix. The main variant of LEGO aims to fit a given set of pairwise distances. We used another variant of LEGO that, like OASIS, learns from relative distances. In our experimental setting, the loss is incurred for same-class examples being more than a certain distance away, and different class examples being less than a certain distance away. LEGO uses the LogDet divergence for regularization, as opposed to the Frobenius norm used in OASIS.

For all these approaches, we used an implementation provided by the authors. Algorithms were implemented in Matlab, with runtime bottlenecks implemented in C for speedup (except LEGO). We test below two variants of OASIS applied to the Caltech256 data set: a pure Matlab implementation, and one that has a *C* components. We used a C^{++} implementation of OASIS for the web-scale experiments described below.

We have also experimented with the methods of Xing et al. (2003) and RCA (Bar-Hillel et al., 2003). We found the method of Xing et al. (2003) to be too slow for the sets in our experiments. RCA is based on a per-class eigen decomposition that is not well defined when the number of samples is smaller than the feature dimensionality. We therefore experimented with a preprocessing phase of dimensionality reduction followed by RCA, but results were inferior to other methods and were not included in the evaluations below. RCA also did not perform well when tested on the full data, where dimensionality was not a problem, possibly because it is not designed to handle well sparse data.

5.3.2 EXPERIMENTAL PROTOCOL

We tested all methods on subsets of classes taken from the Caltech256 repository. Each subset was built such that it included semantically diverse categories, spanning the full range of classification difficulty, as measured by Griffin et al. (2007). We used subsets of sizes 10, 20, 50 and 249 classes (we used 249 classes since classes 251-256 are strongly correlated with other classes, and since class 129 did not contain enough large images). The full lists of categories in each set are given in Appendix B. For each set, images from each class were split into a training set of 40 images and a test set of 25 images, as proposed by Griffin et al. (2007).

We used cross-validation to select the values of hyper parameters for all algorithms except MCML. Models were learned on 80% of the training set (32 images), and evaluated on the remaining 20%. Cross validation was used for setting the following hyper parameters: the early stopping time for OASIS; the ω parameter for LMNN ($\omega \in \{0.125, 0.25, 0.5\}$), and the regularization parameter η for LEGO ($\eta \in \{0.02, 0.08, 0.32\}$). We found that LEGO was usually not sensitive to the choice of η, yielding a variance that was smaller than the variance over different cross-validation splits. Results reported below were obtained by selecting the best value of the hyper parameter and then training again on the full training set (40 images). For MCML, we used the default parameters supplied with the code from the authors, since its very long run time and multiple parameters made it non-feasible to tune hyper parameters on this data.

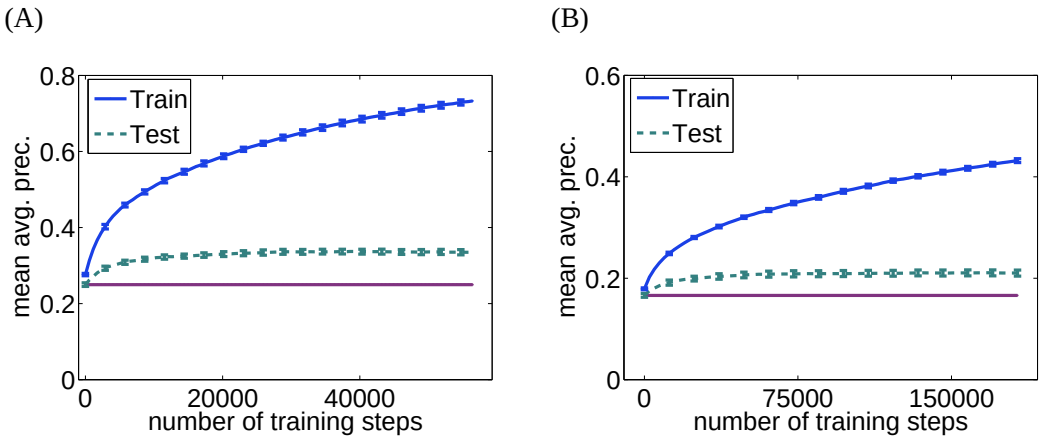


Figure 5: Mean average precision of OASIS as a function of the number of training steps. Error bars represent standard error of the mean over 5 selections of training (40 images) and test (25 images) sets. Performance is compared with a baseline obtained using the naïve Euclidean metric on the feature vector. C=0.1 **(A)** 10 classes. Test performance saturates around 30K training steps, while going over all triplets would require 2.8 million steps. **(B)** 20 classes.

5.3.3 RESULTS

Figure 5 traces the mean average precision over the training and the test sets as it progresses during learning. For the 10 classes task, precision on the test set saturates early (around 35K training steps), and then decreases very slowly.

Figure 6 and Table 4 compare the precision obtained with OASIS, with four competing approaches, as described above (Section 5.3.1). OASIS achieved consistently superior results throughout the full range of k (number of neighbors) tested, and on all four sets studied. Interestingly, we found that LMNN performance on the training set was often high, suggesting that it overfits the training set. This behavior was also noted by Weinberger et al. (2006) in some of their experiments.

OASIS achieves superior or equal performance, with a runtime that is faster by about two orders of magnitudes than MCML, and about one order of magnitude faster than LMNN. The run time of OASIS and LEGO was measured until the point of early stopping.

Table 5 shows the total CPU time in minutes for training each of the algorithms compared (measured on a standard 1.8GHz Intel Xeon CPU). For the purpose of a fair comparison with competing approaches, we tested two implementations of OASIS: The first was fully implemented Matlab. The second had the core of the algorithm implemented in C and called from Matlab.[3] LMNN code and MCML code were supplied by the authors and implemented in Matlab, with core parts implemented in C. LEGO code was supplied by the authors and fully implemented in Matlab.

Importantly, we found that Matlab does not make full use of the speedup that can be gained by sparse image representation. As a result, the C/C^{++} implementation of OASIS that we tested is significantly faster.

3. The OASIS code is available online at *http://ai.stanford.edu/∼gal/Research/OASIS*

10 classes	OASIS Matlab	MCML Matlab+C	LEGO Matlab	LMNN Matlab+C	Euclidean -
Mean avg prec	**33** $\pm$ 1.6	29 $\pm$ 1.7	27 $\pm$ 0.8	24 $\pm$ 1.6	23 $\pm$ 0.9
Top 1 prec.	**43** $\pm$ 4.0	39 $\pm$ 5.1	39 $\pm$ 4.8	38 $\pm$ 5.4	37 $\pm$ 4.1
Top 10 prec.	**38** $\pm$ 1.3	33 $\pm$ 1.8	32 $\pm$ 1.2	29 $\pm$ 2.1	27 $\pm$ 1.5
Top 50 prec.	**23** $\pm$ 1.5	22 $\pm$ 1.3	20 $\pm$ 0.5	18 $\pm$ 1.5	18 $\pm$ 0.7
20 classes	OASIS	MCML	LEGO	LMNN	Euclidean
Mean avg. prec	**21** $\pm$ 1.4	17 $\pm$ 1.2	16 $\pm$ 1.2	14 $\pm$ 0.6	14 $\pm$ 0.7
Top 1 prec.	**29** $\pm$ 2.6	26 $\pm$ 2.3	26 $\pm$ 2.7	26 $\pm$ 3.0	25 $\pm$ 2.6
Top 10 prec.	**24** $\pm$ 1.9	21 $\pm$ 1.5	20 $\pm$ 1.4	19 $\pm$ 1.0	18 $\pm$ 1.0
Top 50 prec.	**15** $\pm$ 0.4	14 $\pm$ 0.5	13 $\pm$ 0.6	11 $\pm$ 0.2	12 $\pm$ 0.2
50 classes	OASIS	MCML	LEGO	LMNN	Euclidean
Mean avg. prec.	**12** $\pm$ 0.4	$*$	9 $\pm$ 0.4	8 $\pm$ 0.4	9 $\pm$ 0.4
Top 1 prec.	**21** $\pm$ 1.6	$*$	18 $\pm$ 0.7	18 $\pm$ 1.3	17 $\pm$ 0.9
Top 10 prec.	**16** $\pm$ 0.4	$*$	13 $\pm$ 0.6	12 $\pm$ 0.5	13 $\pm$ 0.4
Top 50 prec.	**10** $\pm$ 0.3	$*$	8 $\pm$ 0.3	7 $\pm$ 0.2	8 $\pm$ 0.3

Table 4: Mean average precision and precision at top 1, 10, and 50 of all compared methods. Values are averages over 5 cross validation folds; $\pm$ values are the standard deviation across the 5 folds. A '*' denotes cases where a method took more than 5 days to converge.

classes	OASIS Matlab	OASIS Matlab+C	MCML Matlab+C	LEGO Matlab	LMNN (naive) Matlab+C	fast-LMNN Matlab+C
10	42 $\pm$ 15	0.12 $\pm$ .03	1835 $\pm$ 210	143 $\pm$ 44	337 $\pm$ 169	247 $\pm$ 209
20	45 $\pm$ 8	0.15 $\pm$ .02	7425 $\pm$ 106	533 $\pm$ 49	631 $\pm$ 40	365 $\pm$ 62
50	25 $\pm$ 2	1.6 $\pm$ .04	$*$	711 $\pm$ 28	960 $\pm$ 80	2109 $\pm$ 67
249	485 $\pm$ 113	1.13 $\pm$ .15	$*$	$**$	$**$	$**$

Table 5: Runtime (minutes) of all compared methods. Values are averages over 5 cross validation folds, $\pm$ values are the standard deviation across the 5 folds. A '$*$' denotes cases where a method took more than 5 days to converge. A '$**$' denotes cases where performance was worse than the Euclidean baseline.

5.4 Parallel Training

We presented OASIS as optimizing an objective function at each step. Since OASIS is based on the PA framework, it is also known to minimize a global objective of the form

$$\|\mathbf{W}\|_{Fro}^2 + C\sum_i l_i$$

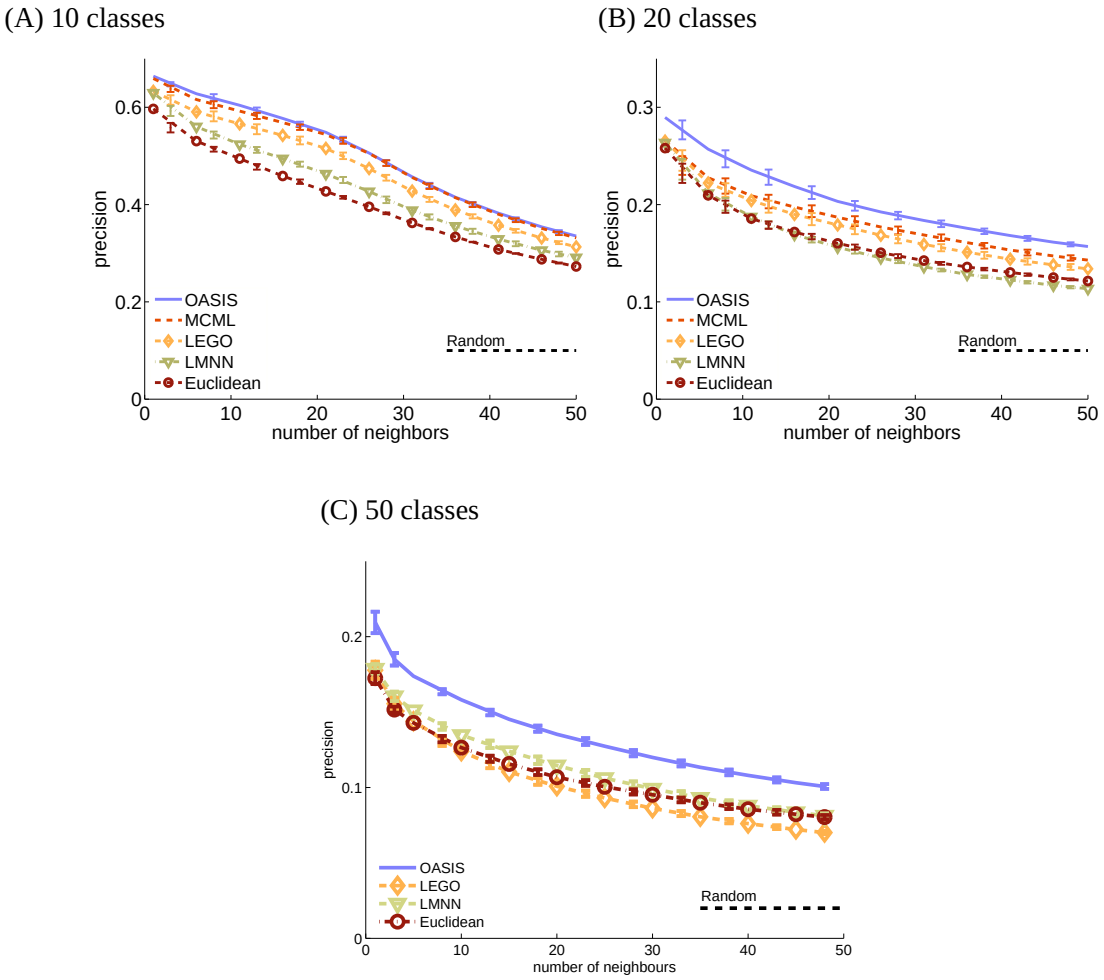


Figure 6: Comparison of the performance of OASIS, LMNN, MCML, LEGO and the Euclidean metric in feature space. Each curve shows the precision at top k as a function of k neighbors. The results are averaged across 5 train/test partitions (40 training images, 25 test images), error bars are standard error of the means (s.e.m.), black dashed line denotes chance performance. **(A)** 10 classes. **(B)** 20 classes. **(C)** 50 classes.

as shown by Crammer et al. (2006) This objective is convex since the losses l_i are linear in $\mathbf{W}$. For such convex functions, it is guaranteed that any linear combination of solutions is superior than each of the individual solutions. This property suggests another way to speed up training, by training multiple rankers in parallel and averaging the resulting models. Each of the individual models can be trained with a smaller number of iterations. Note however that there is no guarantee that the total CPU time is improved.

Figure 7 demonstrates this approach; we trained 5 or 10 rankers in parallel and plot the test set mean average precision as a function of the number of training iterations.

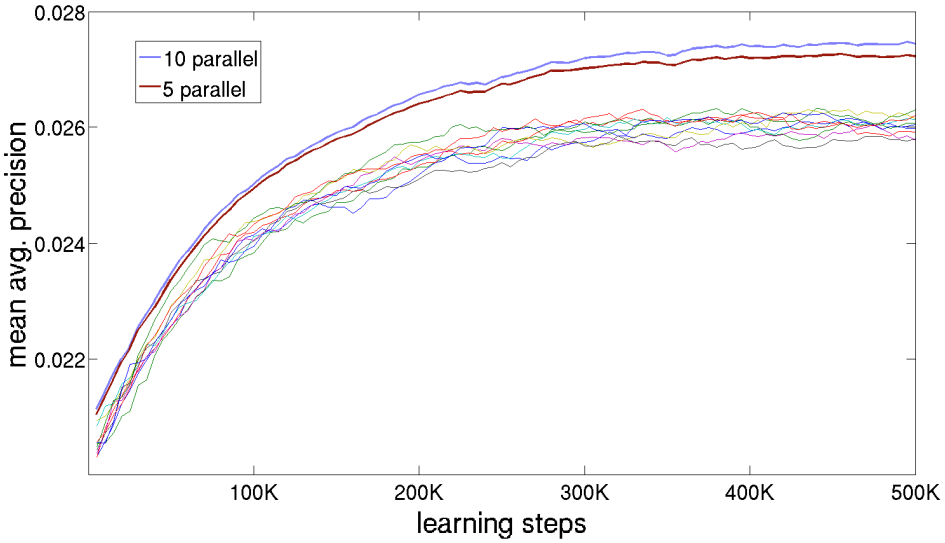


Figure 7: Comparing individual rankers and a linear combination of 5 and 10 rankers. Results are for an experiment with 249 classes of the Caltech256 data set.

.

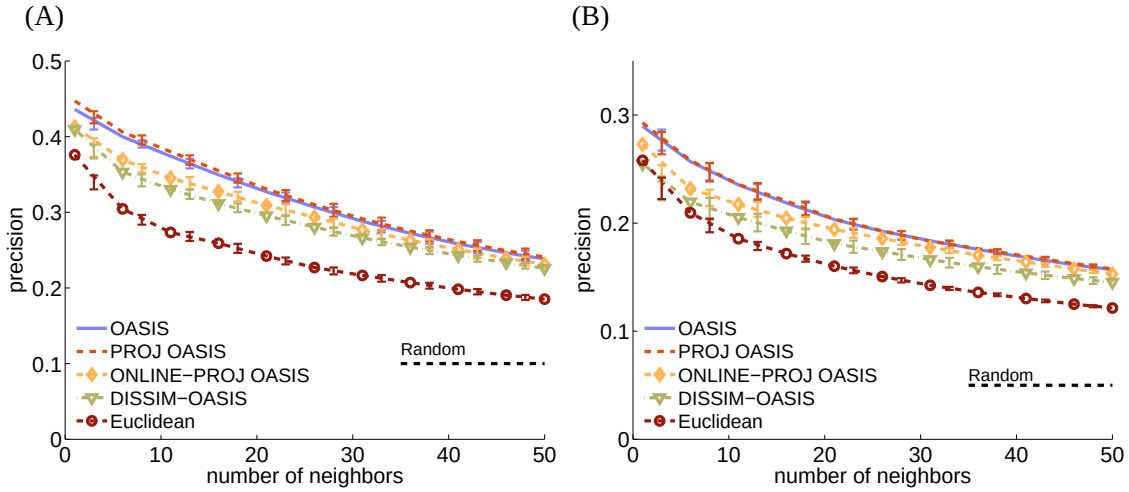


Figure 8: Comparison of Symmetric variants of OASIS. **(A)** 10 classes. **(B)** 20 classes.

6. Symmetry and Positivity

The similarity matrix **W** learned by OASIS is not guaranteed to be positive or even symmetric. Some applications, like ranking images by semantic relevance to a given image query are known to be non-symmetric when based on human judgement (Tversky, 1977). However, in some applications symmetry or positivity constraints reflect a prior knowledge that may help avoiding overfitting.

Furthermore positive $\mathbf{W}$ impose a Mahalanobis metric over the data, that can be further factorized to extract a linear projection of the data into a Euclidean space: $\mathbf{x}^T\mathbf{W}\mathbf{y} = (\mathbf{A}\mathbf{x})^T(\mathbf{A}\mathbf{y})$ such that $\mathbf{A}^T\mathbf{A} = \mathbf{W}$. Such projection $\mathbf{A}$ of the data can be useful for visualization and exploratory analysis of data for example in scientific applications. We now discuss variants of OASIS that learn a symmetric or positive matrices.

6.1 Symmetric Similarities

A simple approach to enforce symmetry is to project the OASIS model $\mathbf{W}$ onto the set of symmetric matrices $\mathbf{W}' = sym(\mathbf{W}) = \frac{1}{2}\left(\mathbf{W}^T + \mathbf{W}\right)$. The update procedure then consists of a series of gradient steps followed by projection to the feasible set (of symmetric matrices). This approach is sometimes called projected gradient, and we denote it here *Online-Proj-Oasis*. Alternatively, projection can also be applied after learning is completed (denoted here *Proj-Oasis*).

Alternatively, the asymmetric score function $S_{\mathbf{W}}(p_i, p_j)$ in the loss l_W can be replaced with a symmetric score

$$S'_{\mathbf{W}}(p_i, p_j) \equiv -(p_i - p_j)^T \ \mathbf{W}\ (p_i - p_j)\ .$$

and derive an OASIS-like algorithm (which we call *Dissim-Oasis*). The optimal update for this loss has a symmetric gradient $\mathbf{V}'^i = (p_i - p_i^+)(p_i - p_i^+)^T - (p_i - p_i^-)(p_i - p_i^-)^T$. Therefore, if $\mathbf{W}^0$ is initialized with a symmetric matrix (for example, the identity matrix) all $\mathbf{W}^i$ are guaranteed to remain symmetric. *Dissim-Oasis* is closely related to LMNN (Weinberger et al., 2006). This can be seen be casting the batch objective of LMNN, into an online setup, which has the form $err(W) = -\omega \cdot S'_{\mathbf{W}}(p_i, p_i^+) + (1-\omega) \cdot l'_{\mathbf{W}}(p_i, p_i^+, p_i^-)$. This online version of LMNN becomes equivalent to *D*issim-Oasis for $\omega = 0$.

Figure 8 compares the precision of the different symmetric methods with the original OASIS. All symmetric variants performed slightly worse, or equal to the original asymmetric OASIS. Asymmetric OASIS is also twice faster than DISSIM-OASIS. The precision of *Proj-Oasis* was equivalent to that of OASIS. This was because the asymmetric OASIS learning rule actually converged to an almost-symmetric model (as measured by a symmetry index $\rho(\mathbf{W}) = \frac{\|sym(\mathbf{W})\|_2}{\|\mathbf{W}\|_2} = 0.94$).

6.2 Positive Similarity

Most similarity learning approaches focus on learning metrics. In the context of OASIS, when $\mathbf{W}$ is positive semi definite (PSD), it defines a Mahalanobis distance over the images. The matrix square-root of $\mathbf{W}$, $\mathbf{A}^T\mathbf{A} = \mathbf{W}$ can then be used to project the data into a new space in which the Euclidean distance is equivalent to the $\mathbf{W}$ distance in the original space.

We experimented with positive variants of OASIS, where we repeatedly projected the learned model onto the set of PSD matrices, once every t iterations. Projection is done by taking the eigen decomposition $\mathbf{W} = \mathbf{V} \cdot \mathbf{D} \cdot \mathbf{V}^T$ where $\mathbf{V}$ is the eigenvector matrix and $\mathbf{D}$ is the diagonal eigenvalues matrix limited to positive eigenvalues. Figure 9 traces precision on the test set throughout learning for various values of t.

The effect of positive projections is complex. First, continuously projecting once every few steps helps to reduce overfitting, as can be observed by the slower decline of the blue curve (upper smooth curve) compared to the orange curve (lowest curve). However, when projection is performed after many steps (instead of continuously), performance of the projected model actually outperforms the continuous-projection model (upper jittery curve). The reason for this effect is likely to be that the

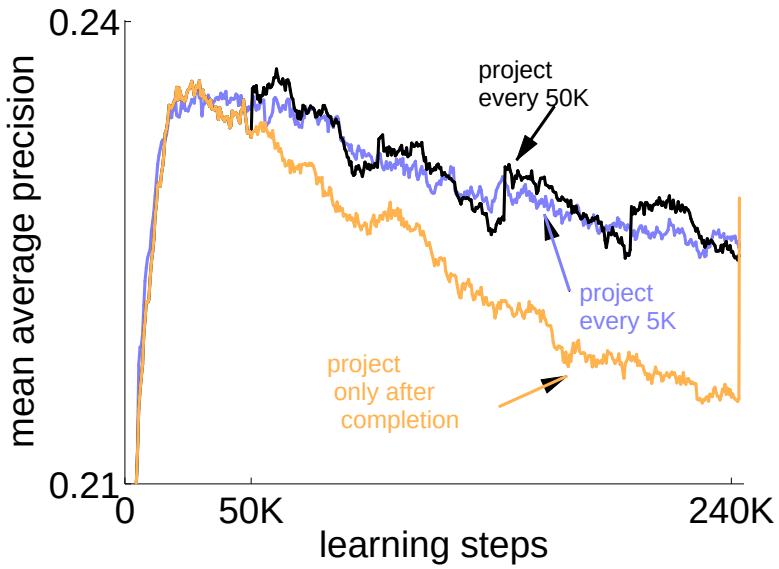


Figure 9: Mean average precision (mAP) during training for three PSD projection schemes, using the set of 20 classes from caltech256.

estimates of the positive sub-space are very noisy when only based on a few samples (see also Chen et al. 2009, Section 2.1). Indeed, accurate estimation of the negative subspace is known to be a hard problem, because small perturbations can turn a negative but small eigenvalue, into a small but positive one. As a result, the set of vectors selected based on having positive eigenvalues, is highly variable. We found that this effect was so strong, that the optimal projection strategy is to avoid projection throughout learning completely. Instead, projecting into PSD after learning (namely, after a model was chosen using early stopping) provided the best performance in our experiments.

An interesting alternative to obtain a PSD matrix was explored by Kulis et al. (2009) and Jain et al. (2008a). Using a LogDet divergence between two matrices $D_{ld}(X,Y) = tr(XY^{-1}) - log(det(XY^{-1}))$ ensures that, given an initial PSD matrix, all subsequent matrices will be PSD as well. It would be interesting to test the effect of using LogDet regularization in the OASIS setup.

7. Discussion

We have presented OASIS, a scalable algorithm for learning image similarity that captures both semantic and visual aspects of image similarity. Three key factors contribute to the scalability of OASIS. First, using a large margin online approach allows training to converge even after seeing a small fraction of potential pairs. Second, the objective function of OASIS does not require the similarity measure to be necessarily a metric during training, although it appears to naturally converge to a symmetric solution. Finally, we use a sparse representation of low level features which allows computing scores very efficiently.

We found that OASIS performs well in a wide range of scales: from problems with thousands of images, where it slightly outperforms existing metric-learning approaches, to large web-scale problems, where it achieves high accuracy, as estimated by human evaluators.

OASIS differs from previous methods in that the similarity measure that it learns is not forced to be a metric, or even symmetric. When the number of available samples is small, it is useful to add constraints that reflect prior knowledge on the type of similarity measure expected to be learned. However, we found that these constraints were not helpful even for problems with a few hundreds of samples. Interestingly, human judgements of pairwise similarity are known to be asymmetric, a property that can be easily captured by an OASIS model.

OASIS learns a class-independent model: it is not aware of which queries or categories were shared by two similar images. As such, it is more limited in its descriptive power and it is likely that class-dependent similarity models could improve precision. On the other hand, class-independent models could generalize to handle classes that were not observed during training, as in transfer learning. Large scale similarity learning, applied to images from a large variety of classes, could therefore be a useful tool to address real-world problems with a large number of classes.

Acknowledgments

This work was supported by the Israeli Science Foundation (ISF 1001/08). We thank Andrea Frome for very helpful discussions and comments on the manuscript. We thank Amir Globerson, Killian Weinberger and Prateek Jain, each providing an implementation of their method for our experiments.

Appendix A. Human Evaluation

The following text was given as instructions to human evaluators when judging the relevance of images to a query image.

```
Scenario:
     A user is searching images to use in a presentation he/she plans to
  give.  The user runs a standard image search, and selects an image,
  the ‘‘query image’’.  The user then wishes to refine the search and
  look for images that are SEMANTICALLY similar to the query image.
     The difficulty lies, in the definition of ‘‘SEMANTICALLY’’. This can
  have many interpretations, and you should take that into account.
     So for instance, if you see an image of a big red truck, you can
  interpret the user intent (the notion of semantically similar) in
  various ways:
    - any big red truck
    - any red truck
    - any big truck
    - any truck
    - any vehicle
  You should interpret ‘‘SEMANTICALLY’’ in a broad sense rather than
  in a strict sense but feel free to draw the line yourself (although
```

```
  be consistent).

Your task:
     You will see a set of query images on the left side of the screen,
  and a set of potential candidate matches, 5 per row, on the
  right. Your job is to decide for each of the candidate images if it
  is a good semantic match to the query image or not. The default is
  that it is NOT a good match. Furthermore, if for some reason you
  cannot make-up your mind, then answer ''can't say''.
```

Appendix B. Caltech256 Class Sets

- **10 classes**: *bear, skyscraper, billiards, yo-yo, minotaur, roulette-wheel, hamburger, laptop-101, hummingbird, blimp.*
- **20 classes***: airplanes-101, mars, homer-simpson, hourglass, waterfall, helicopter-101, mountain-bike starfish-101, teapot, pyramid, refrigerator, cowboy-hat, giraffe, joy-stick, crab-101, bird-bath, fighter-jet tuning-fork, iguana, dog.*
- **50 classes***: car-side-101, tower-pisa, hibiscus, saturn, menorah-101, rainbow, cartman, chandelier-101, backpack, grapes, laptop-101, telephone-box, binoculars, helicopter-101, paper-shredder, eiffel-tower, top-hat, tomato, star-fish-101, hot-air-balloon, tweezer, picnic-table, elk, kangaroo-101, mattress, toaster, electric-guitar-101, bathtub, gorilla, jesus-christ, cormorant, mandolin, light-house, cake, tricycle, speed-boat, computer-mouse, superman, chimp, pram, fried-egg, fighter-jet, unicorn, greyhound, grasshopper, goose, iguana, drinking-straw, snake, hot-dog.*
- **249 classes***: classes 1-250, excluding class 129 (leopards-101), which had less than 65 large enough images.*

References

A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proc. of 20th International Conference on Machine Learning (ICML)*, page 11, 2003.

L. Bottou. Large-scale machine learning and stochastic algorithms. In *NIPS 2008 Workshop on Optimization for Machine Learning*, 2008.

Y. Chen, E.K. Garcia, M.R. Gupta, A. Rahimi, and L. Cazzanti. Similarity-based classification: Concepts and algorithms. *The Journal of Machine Learning Research*, 10:747–776, 2009.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research (JMLR)*, 7:551–585, 2006.

J.V. Davis, B. Kulis, P. Jain, S. Sra, and I.S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM Press New York, NY, USA, 2007.

P. Duygulu, K. Barnard, N. de Freitas, and D. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *European Conference on Computer Vision (ECCV)*, pages 97–112, 2002.

S.L. Feng, R. Manmatha, and V. Lavrenko. Multiple Bernoulli relevance models for image and video annotation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *International Conference on Computer Vision*, pages 1–8, 2007.

A. Globerson and S. Roweis. Metric learning by collapsing classes. *Advances in Neural Information Processing Systems*, 18:451, 2006.

D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(8):1371–1384, 2008.

D. Grangier, F. Monay, and S. Bengio. Learning to retrieve images from text queries with a discriminative model. In *International Conference on Adaptive Multimedia Retrieval (AMR)*, 2006.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2006.

P. Jain, B. Kulis, I. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In *Advances in Neural Information Processing Systems*, volume 22, 2008a.

P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008b.

J. Jeon and R. Manmatha. Using maximum entropy for automatic image annotation. In *International Conference on Image and Video Retrieval*, pages 24–32, 2004.

B. Kulis, M.A. Sustik, and I.S. Dhillon. Low-rank kernel learning with bregman matrix divergences. *Journal of Machine Learning Research*, 10:341–376, 2009.

G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research (JMLR)*, 5:27–72, 2004.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

W.S. Noble. Multi-kernel learning for biology. In *NIPS 2008 workshop on kernel learning*, 2008.

T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(7):971–987, 2002.

P. Quelhas, F. Monay, J. M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. J. Van Gool. Modeling scenes with local descriptors and latent aspects. In *International Conference on Computer Vision*, pages 883–890, 2005.

N. Rasiwasia and N. Vasconcelos. A study of query by semantic example. In *3rd International Workshop on Semantic Learning and Applications in Multimedia*, 2008.

R. Rosales and G. Fung. Learning sparse metrics via linear programming. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 367–373. ACM New York, NY, USA, 2006.

M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*. Bradford Book, 2004.

V. Takala, T. Ahonen, and M. Pietikainen. Block-based methods for image retrieval using local binary patterns. In *Scandinavian Conference on Image Analysis (SCIA)*, 2005.

K. Tieu and P. Viola. Boosting image retrieval. *International Journal of Computer Vision (IJCV)*, 56(1):17 – 36, 2004.

A. Torralba, R. Fergus, and W. T. Freeman. Tiny images. Technical Report MIT-CSAIL-TR-2007-024, Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, 2007. URL `http://dspace.mit.edu/handle/1721.1/37291`.

A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. *Advances in Neural Information Processing Systems*, 18:1473, 2006.

K.Q. Weinberger and L.K. Saul. Fast solvers and efficient implementations for distance metric learning. In *ICML25*, pages 1160–1167, 2008.

E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 521–528, Cambridge, MA, 2003. MIT Press.

L. Yang. Distance metric learning: A comprehensive survey. Technical report, Michigan State University, 2006.

2.2 Online Learning in the Embedded Manifold of Low-rank Matrices

Journal of Machine Learning Research 13 (2012) 429-458 Submitted 6/11; Revised 11/11; Published 2/12

Online Learning in the Embedded Manifold of Low-rank Matrices

Uri Shalit[*] URI.SHALIT@MAIL.HUJI.AC.IL
Computer Science Department and ICNC/ELSC
The Hebrew University of Jerusalem
91904 Jerusalem, Israel

Daphna Weinshall DAPHNA@CS.HUJI.AC.IL
Computer Science Department
The Hebrew University of Jerusalem
91904 Jerusalem, Israel

Gal Chechik[†] GAL@GOOGLE.COM
The Gonda Brain Research Center
Bar Ilan University
52900 Ramat-Gan, Israel

Editor: Léon Bottou

Abstract

When learning models that are represented in matrix forms, enforcing a low-rank constraint can dramatically improve the memory and run time complexity, while providing a natural regularization of the model. However, naive approaches to minimizing functions over the set of low-rank matrices are either prohibitively time consuming (repeated singular value decomposition of the matrix) or numerically unstable (optimizing a factored representation of the low-rank matrix). We build on recent advances in optimization over manifolds, and describe an iterative online learning procedure, consisting of a gradient step, followed by a *second-order retraction* back to the manifold. While the ideal retraction is costly to compute, and so is the projection operator that approximates it, we describe another retraction that can be computed efficiently. It has run time and memory complexity of $O((n+m)k)$ for a rank-k matrix of dimension $m \times n$, when using an online procedure with rank-one gradients. We use this algorithm, LORETA, to learn a matrix-form similarity measure over pairs of documents represented as high dimensional vectors. LORETA improves the mean average precision over a passive-aggressive approach in a factorized model, and also improves over a full model trained on pre-selected features using the same memory requirements. We further adapt LORETA to learn positive semi-definite low-rank matrices, providing an online algorithm for *low-rank metric learning*. LORETA also shows consistent improvement over standard weakly supervised methods in a large (1600 classes and 1 million images, using *ImageNet*) multi-label image classification task.

Keywords: low rank, Riemannian manifolds, metric learning, retractions, multitask learning, online learning

1. Introduction

Many learning problems involve models represented in matrix form. These include metric learning, collaborative filtering, and multi-task learning where all tasks operate over the same set of features.

∗. Also at The Gonda Brain Research Center, Bar Ilan University, 52900 Ramat-Gan, Israel.
†. Also at Google Research, 1600 Amphitheatre Parkway, Mountain View CA, 94043.

In many of these tasks, a natural way to regularize the model is to limit the rank of the corresponding matrix. In metric learning, a low-rank constraint allows to learn a low dimensional representation of the data in a discriminative way. In multi-task problems, low-rank constraints provide a way to tie together different tasks. In all cases, low-rank matrices can be represented in a factorized form that dramatically reduces the memory and run-time complexity of learning and inference with that model. Low-rank matrix models could therefore scale to handle substantially many more features and classes than models with full rank dense matrices.

Unfortunately, the rank constraint is non-convex, and in the general case, minimizing a convex function subject to a rank constraint is NP-hard (Natarajan, 1995).[1] As a result of these issues, two main approaches have been commonly used to address the problem of learning under a low-rank constraint. Sometimes, a matrix $W \in \mathbb{R}^{n\times m}$ of rank k is represented as a product of two low dimension matrices $W = AB^T, A \in \mathbb{R}^{n\times k}, B \in \mathbb{R}^{m\times k}$ and simple gradient descent techniques are applied to each of the product terms separately (Bai et al., 2009). Second, projected gradient algorithms can be applied by repeatedly taking a gradient step and projecting back to the manifold of low-rank matrices. Unfortunately, computing the projection to that manifold becomes prohibitively costly for large matrices and cannot be computed after every gradient step.

Work in the field has focused mostly on two realms. First, learning low-rank positive semi-definite (PSD) models (as opposed to general low-rank models), as in the works of Kulis et al. (2009) and Meyer et al. (2011). Second, approximating a noisy matrix of observations by a low-rank matrix, as in the work of Negahban and Wainwright (2010). This task is commonly addressed in the field of recommender systems. Importantly, the current paper does not address the problem of low-rank *approximation to a given data matrix*, but rather addresses the problem of learning a *low-rank parametric model* in the context of ranking and classification.

In this paper we propose new algorithms for online learning on the manifold of low-rank matrices. It is based on an operation called *retraction*, which is an operator that maps from a vector space that is tangent to the manifold, into the manifold (Do Carmo, 1992; Absil et al., 2008). Retractions include the projection operator as a special case, but also include other operators that can be computed substantially more efficiently. We use second order retractions to develop LORETA —an online algorithm for learning low-rank matrices. LORETA has a memory and run time complexity of $O((n+m)k)$ per update when the gradients have rank one. We show below that the case of rank-one gradients is relevant to numerous online learning problems.

We test LORETA in two different domains and learning tasks. First, we learn a bilinear similarity measure among pairs of text documents, where the number of features (text terms) representing each document could become very large. LORETA performed better than other techniques that operate on a factorized model, and also improves retrieval precision by 33% as compared with training a full rank model over pre-selected most informative features, using comparable memory footprint. Second, we applied LORETA to image multi-label ranking, a problem in which the number of classes could grow to millions. LORETA significantly improved over full rank models, using a fraction of the memory required. These two experiments suggest that low-rank optimization could become very useful for learning in high-dimensional problems.

1. Some special cases are solvable (notably, PCA), relying mainly on singular value decomposition (Fazel et al., 2005) and semi-definite programming techniques. For SDP of rank $k \geq 2$ it is not known whether this problem is NP-hard. For $k = 1$ it is equivalent to the MAX-CUT problem (Briët et al., 2010). Both SDP and SVD scale poorly to large scale tasks.

This paper is organized as follows. We start with an introduction to optimization on manifolds, describing the notion of retractions. We then derive our low-rank online learning algorithm in three variants: one which learns a general low-rank matrix, one which learns a low-rank PSD matrix, and one which concentrates most of the learning in a reduced dimensional space. Finally we test our algorithms in two applications: learning similarity of text documents, and multi-label ranking on a set of one million images.

This paper extends a shorter version published in Advances in Neural Information Systems (Shalit et al., 2010), by adding a new PSD version of the algorithm, much larger-scale and wider experiments, giving a full mathematical discussion and proofs, and adding thorough complexity analysis.

2. Optimization on Riemannian Manifolds

The field of numerical optimization on smooth manifolds has advanced significantly in the past few years. For a recent exposition on this subject see Absil et al. (2008). We start with a short introduction to embedded manifolds, which are the focus of this paper.

An *embedded manifold* is a smooth subset of an ambient space $\mathbb{R}^n$. For instance, the set $\{\mathbf{x} : ||\mathbf{x}||_2 = 1, \mathbf{x} \in \mathbb{R}^n\}$, the unit sphere, is an $n-1$ dimensional manifold embedded in n-dimensional space $\mathbb{R}^n$. As another example, the *orthogonal group* O_n, which comprises of the set of orthogonal $n \times n$ matrices, is an $\frac{n(n-1)}{2}$ dimensional manifold embedded in $\mathbb{R}^{n\times n}$. Here we focus on the manifold of *low-rank* matrices, namely, the set of $n \times m$ matrices of rank k where $k < m, n$. It is an $(n+m)k - k^2$ dimensional manifold embedded in $\mathbb{R}^{n\times m}$, which we denote $\mathcal{M}_k^{n,m}$, or plainly $\mathcal{M}$. Embedded manifolds inherit many properties from the ambient space, a fact which simplifies their analysis. For example, the natural Riemannian metric for embedded manifolds is simply the Euclidean metric restricted to the manifold.

Motivated by online learning, we focus here on developing a stochastic gradient descent procedure to minimize a loss function $\mathcal{L}$ over the manifold of low-rank matrices $\mathcal{M}_k^{n,m}$,

$$\min_W \quad \mathcal{L}(W) \qquad \text{s.t.} \quad W \in \mathcal{M}_k^{n,m} \quad .$$

To illustrate the challenge in this problem, consider a simple stochastic gradient descent algorithm (Figure 1). At every step t of the algorithm, a gradient step update $W^t - \tilde{\nabla}\mathcal{L}(W^t)$ takes the model outside of the manifold $\mathcal{M}$ and has to be mapped back onto the manifold. The most common mapping operation is the *projection* operation, which, given a point $W^t - \tilde{\nabla}\mathcal{L}(W^t)$ outside the manifold, would find the closest point in $\mathcal{M}$. Unfortunately, the projection operation is very expensive to compute for the manifold of low-rank matrices, since it basically involves a singular value decomposition. Here we describe a wider class of operations called *retractions*, that serve a similar purpose: they find a point on the manifold that is in the direction of the gradient. To explain how retractions are computed, we first describe the notion of a *tangent space* and the *Riemannian gradient* of a function on a manifold.

2.1 Riemannian Gradient and the Tangent Space

Each point W in an embedded manifold $\mathcal{M}$ has a tangent space associated with it, denoted $T_{\mathbf{W}}\mathcal{M}$, as shown in Figure 2 (for a formal definition of the tangent space, see Appendix A). The tangent space is a vector space of the same dimension as the manifold that can be identified in a natural way

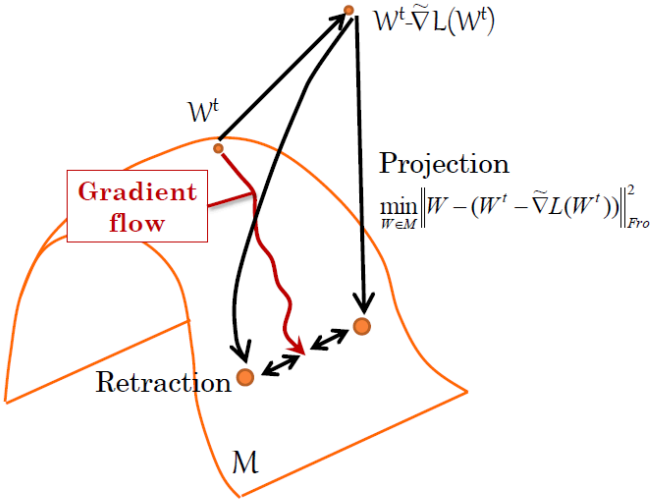


Figure 1: Projection onto the manifold is just a particular case of a retraction. Retractions are defined as operators that approximate the geodesic gradient flow on the manifold.

with a linear subspace of the ambient space. It is usually simple to compute the linear projection P_W of any point in the ambient space onto the tangent space $T_{\mathbf{W}}\mathcal{M}$.

Given a manifold $\mathcal{M}$ and a differentiable function $\mathcal{L} : \mathcal{M} \to \mathbb{R}$, the *Riemannian gradient* $\nabla\mathcal{L}(W)$ of $\mathcal{L}$ on $\mathcal{M}$ at a point $\mathbf{W}$ is a vector in the tangent space $T_{\mathbf{W}}\mathcal{M}$. A very useful property of embedded manifolds is the following: given a differentiable function f defined on the ambient space (and thus on the manifold), the Riemannian gradient of f at point W is simply the linear projection P_W of the Euclidean gradient of f onto the tangent space $T_W\mathcal{M}$.

Thus, if we denote the Euclidean gradient of $\mathcal{L}$ in $\mathbb{R}^{n\times m}$ by $\tilde{\nabla}\mathcal{L}$, we have $\nabla\mathcal{L}(W) = P_W(\tilde{\nabla}\mathcal{L})$. An important consequence follows in case the manifold represents the set of points obeying a certain constraint. In this case the Riemannian gradient of f is equivalent to the Euclidean gradient of f minus the component which is normal to the constraint. Indeed this normal component is exactly the component which is irrelevant when performing constrained optimization.

The Riemannian gradient allows us to compute $W^{t+\frac{1}{2}} = W^t - \eta^t\nabla\mathcal{L}(W)$, for a given iterate point W^t and step size η^t. We now examine how $W^{t+\frac{1}{2}}$ can be mapped back onto the manifold.

2.2 Retractions

Intuitively, *retractions* capture the notion of "going along a straight line" on the manifold. The mathematically ideal retraction is called the *exponential mapping* (Do Carmo, 1992, Chapter 3): it maps the tangent vector $\xi \in T_W\mathcal{M}$ to a point along a geodesic curve which goes through W in the direction of ξ Figure 1. Unfortunately, for many manifolds (including the low-rank manifold considered here) calculating the geodesic curve is computationally expensive (Vandereycken et al., 2009). A major insight from the field of Riemannian manifold optimization is that one can use retractions which merely approximate the exponential mapping. Using such retractions maintains the conver-

gence properties obtained with the exponential mapping, but is much cheaper computationally for a suitable choice of mapping.

Definition 1 *Given a point W in an embedded manifold $\mathcal{M}$, a retraction is any function $R_W : T_{\mathbf{W}}\mathcal{M} \to \mathcal{M}$ which satisfies the following two conditions (Absil et al., 2008, Chapter 4):*

1. *Centering: $R_W(0) = W$.*
2. *Local rigidity: The curve $\gamma : (-\varepsilon, \varepsilon) \to \mathcal{M}$ defined by $\gamma_\xi(\tau) = R_W(\tau\xi)$ satisfies $\dot{\gamma}_\xi(0) = \xi$, where $\dot{\gamma}$ is the derivative of γ by τ.*

It can be shown that any such retraction approximates the exponential mapping to a first order (Absil et al., 2008). *Second-order retractions*, which approximate the exponential mapping to second order around W, have to satisfy in addition the following stricter condition:

$$P_W\left(\frac{\mathrm{d}R_W(\tau\xi)}{\mathrm{d}\tau^2}|_{\tau=0}\right) = 0,$$

for all $\xi \in T_W\mathcal{M}$, where P_W is the *linear* projection from the ambient space onto the tangent space $T_{\mathbf{W}}\mathcal{M}$. When viewed intrinsically, the curve $R_W(\tau\xi)$ defined by a second-order retraction has zero acceleration at point W, namely, its second order derivatives are all normal to the manifold. The best known example of a second-order retraction onto embedded manifolds is the projection operation (Absil and Malick, 2010), which maps a point X to the point $Y \in \mathcal{M}$ which is closest to it in the Frobenius norm. That is, the projection of X onto $\mathcal{M}$ is simply:

$$Proj_{\mathcal{M}}(X) = \underset{Y \in \mathcal{M}}{\operatorname{argmin}} \|X - Y\|_{Fro}$$

Importantly, such projections are viewed here as one type of a second order approximation to the exponential mapping, which can be replaced by any other second-order retractions, when computing the projection is too costly (see Figure 1).

Given the tangent space and a retraction, we now define a Riemannian gradient descent procedure for the loss $\mathcal{L}$ at point $W^t \in \mathcal{M}$. Conceptually, the procedure has three steps (Figure 2):

1. **Step 1: Ambient gradient:** Obtain the Euclidean gradient $\tilde{\nabla}\mathcal{L}(W^t)$ in the ambient space.
2. **Step 2: Riemannian gradient:** Linearly project the ambient gradient onto the tangent space $T_{\mathbf{W}}\mathcal{M}$. Compute $\xi^t = P_{W^t}(\tilde{\nabla}\mathcal{L}(W^t))$.
3. **Step 3: Retraction:** Retract the Riemannian gradient ξ^t back to the manifold: $W^{t+1} = R_{W^t}(\xi^t)$.

With a proper choice of step size, this procedure can be proved to have local convergence for any retraction (Absil et al., 2008). In practice, the algorithm merges these three steps for efficiency, as discussed in the next section.

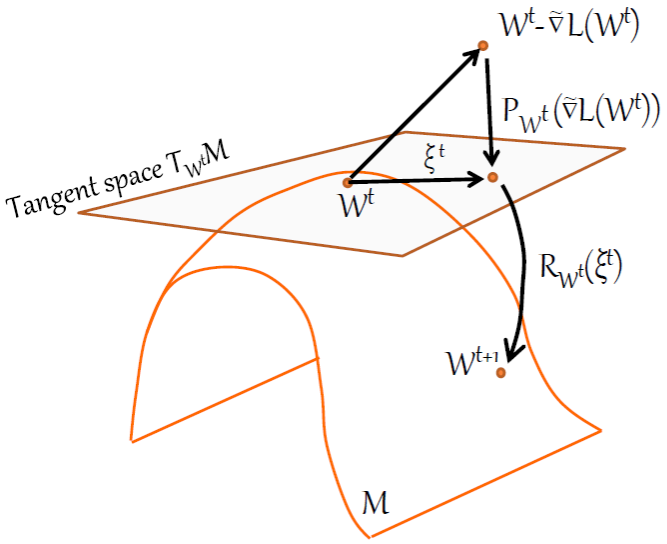


Figure 2: A three step procedure for computing a retracted gradient at point W^t. Step 1: standard (Euclidean) gradient step. Step 2: linearly project ambient gradient onto tangent space $T_{\mathbf{W}}\mathcal{M}$ in order to get the Riemannian gradient ξ^t. Step 3: retract the Riemannian gradient ξ^t back to the manifold.

3. Online Learning on the Low-rank Manifold

Based on the retractions described above, we now present an online algorithm for learning low-rank matrices, by performing stochastic gradient descent on the manifold of low-rank matrices. We name the algorithm LORETA (for a *LOw rank RETraction Algorithm*). At every iteration the algorithm suffers some loss, and performs a Riemannian gradient step followed by a retraction to the manifold $\mathcal{M}_k^{n,m}$. Section 3.1 discusses general online updates. Section 3.2 discusses the very common case where the online updates induce a gradient of rank $r = 1$.

Algorithm 1 : Online algorithm for learning in the manifold of low-rank matrices

Input: Initial low-rank model matrix $W^0 \in \mathcal{M}_k^{n,m}$. Examples $(x_0, x_1, \ldots)$. Loss function $\mathcal{L}$. Gradient descent step sizes $(\eta^0, \eta^1, \ldots)$.

Output: Final low-rank model matrix $W^{final} \in \mathcal{M}_k^{n,m}$.

repeat:

Get example x_t

Calculate the stochastic loss gradient: $\tilde{\nabla}\mathcal{L}(W^t; x_t)$

Linearly project onto the tangent space: $\xi^t = P_{W^t}(\tilde{\nabla}\mathcal{L}(W^t; x_t))$

Retract back to the manifold: $W^{t+1} = R_{W^t}(-\eta^t \xi^t)$

until stopping condition is satisfied

In what follows, lowercase Greek letters like ξ denote an abstract tangent vector, and uppercase Roman letters like A denote concrete matrix representations as kept in memory (taking $n \times m$ float numbers to store). We intermix the two notations, as in $\xi = AZ$, when the meaning is clear from the context. The set of $n \times k$ matrices of rank k is denoted $\mathbb{R}_*^{n \times k}$.

3.1 The General-Rank LORETA Algorithm

In online learning we are repeatedly given a rank-r gradient matrix $Z = \tilde{\nabla} \mathcal{L} W$, and want to compute a step on $\mathcal{M}_k^{n,m}$ in the direction of Z. As a first step we find its linear projection onto the tangent space $\hat{Z} = P_W(Z)$.

We start with a lemma that gives a representation of the tangent space $T_{\mathbf{W}}\mathcal{M}$ (Figure 2), extending the constructions given by Vandereycken and Vandewalle (2010) to the general manifold of low-rank matrices.

Lemma 2 *Let $W \in \mathcal{M}_k^{n,m}$ have a (non-unique) factorization $W = AB^T$, where $A \in \mathbb{R}_*^{n \times k}$, $B \in \mathbb{R}_*^{m \times k}$. Let $A_\perp \in \mathbb{R}^{n \times (n-k)}$ and $B_\perp \in \mathbb{R}^{m \times (m-k)}$ be the orthogonal complements of A and B respectively, such that $A_\perp^T A = 0$, $B_\perp^T B = 0$, $A_\perp^T A_\perp = I_{n-k}$, $B_\perp^T B_\perp = I_{m-k}$. The tangent space to $\mathcal{M}_k^{n,m}$ at W is:*

$$T_{\mathbf{W}}\mathcal{M} = \left\{ \begin{bmatrix} A & A_\perp \end{bmatrix} \begin{bmatrix} M & N_1^T \\ N_2 & 0 \end{bmatrix} \begin{bmatrix} B^T \\ B_\perp^T \end{bmatrix} : M \in \mathbb{R}^{k \times k}, N_1 \in \mathbb{R}^{(m-k) \times k}, N_2 \in \mathbb{R}^{(n-k) \times k} \right\}.$$

Proof The proof is given in Appendix A. ■

We note that the assumption that A and B are both of full column rank is tantamount to assuming that the model W is exactly of rank k, and no less. Let $\xi \in T_{\mathbf{W}}\mathcal{M}$ be a tangent vector to $W = AB^T$. From the characterization above it follows that ξ can be decomposed in a unique manner into three orthogonal components: $\xi = \xi^{AB} + \xi^{AB_\perp} + \xi^{A_\perp B}$, where:

$$\xi^{AB} = AMB^T, \quad \xi^{AB_\perp} = AN_1^T B_\perp^T, \quad \xi^{A_\perp B} = A_\perp N_2 B^T. \tag{1}$$

It is easy to verify that each pair is orthogonal, following from the relations $A_\perp^T A = 0$, $B_\perp^T B = 0$.

We wish to find the three matrices M, N_1 and N_2 associated with $\hat{Z} = P_W(Z)$, such that $\hat{Z} = AMB^T + AN_1^T B_\perp^T + A_\perp N_2 B^T$. We can find each of the matrices M, N_1 and N_2 separately, because each belongs to a space orthogonal to the other two. Thus we solve the following three problems:

$$\begin{aligned} &\operatorname*{arg\,min}_{M \in \mathbb{R}^{k \times k}} && \|Z - AMB^T\|_{Fro}^2, \\ &\operatorname*{arg\,min}_{N_1 \in \mathbb{R}^{(m-k) \times k}} && \|Z - AN_1^T B_\perp^T\|_{Fro}^2, \\ &\operatorname*{arg\,min}_{N_2 \in \mathbb{R}^{(n-k) \times k}} && \|Z - A_\perp N_2 B^T\|_{Fro}^2. \end{aligned}$$

To find the minimum of each of these three equations, we compute the derivatives and set them to zero. The solutions involve the pseudoinverses of A and B. Since A and B are of full column rank, their pseudoinverses are $A^\dagger = (A^T A)^{-1} A^T$, $B^\dagger = (B^T B)^{-1} B^T$.

$$
\begin{aligned}
M &= (A^T A)^{-1} A^T Z B (B^T B)^{-1} = A^\dagger Z B^{\dagger T}, \\
N_1 &= B_\perp^T Z^T A (A^T A)^{-1} = B_\perp^T Z^T A^\dagger, \\
N_2 &= A_\perp^T Z B (B^T B)^{-1} = A_\perp^T Z B^{\dagger T}.
\end{aligned} \tag{2}
$$

The matrix $AA^\dagger$ is the matrix projecting onto the column space of A, and similarly for B. We will denote these matrices by P_A, P_B, etc. For the matrices projecting onto $A_\perp$ and $B_\perp$'s columns we actually have $P_{A_\perp} = A_\perp A_\perp^T$ because the columns of $A_\perp$ are orthogonal, and likewise for $P_{B_\perp}$. Substituting the expressions in *Equation* (2) into expressions of the components of the Riemannian gradient vector in *Equation* (1), we obtain:

$$
\xi^{AB} = P_A Z P_B, \quad \xi^{AB_\perp} = P_A Z P_{B_\perp}, \quad \xi^{A_\perp B} = P_{A_\perp} Z P_B.
$$

We can now define the retraction. The following theorem presents the retraction we will apply.

Theorem 3 *Let $W \in \mathcal{M}_k^{n,m}$, $W = AB^T$, and $W^\dagger = B^{\dagger T} A^\dagger$. Let $\xi \in T_W \mathcal{M}_k^{n,m}$, $\xi = \xi^{AB} + \xi^{AB_\perp} + \xi^{A_\perp B}$, as in Equation* (1)*, and let:*

$$
\begin{aligned}
V_1 &= W + \frac{1}{2}\xi^{AB} + \xi^{A_\perp B} - \frac{1}{8}\xi^{AB} W^\dagger \xi^{AB} - \frac{1}{2}\xi^{A_\perp B} W^\dagger \xi^{AB} \quad , \\
V_2 &= W + \frac{1}{2}\xi^{AB} + \xi^{AB_\perp} - \frac{1}{8}\xi^{AB} W^\dagger \xi^{AB} - \frac{1}{2}\xi^{AB} W^\dagger \xi^{AB_\perp} \quad .
\end{aligned}
$$

The mapping

$$
R_W(\xi) = V_1 W^\dagger V_2
$$

is a second order retraction from a neighborhood $\Theta_W \subset T_W \mathcal{M}_k^{n,m}$ to $\mathcal{M}_k^{n,m}$.

Proof The proof is given in Appendix B. ■

A more succinct representation of this retraction is the following:

Lemma 4 *The retraction $R_W(\xi)$ can be presented as:*

$$
\begin{aligned}
R_W(\xi) = &\left[A\left(I_k + \frac{1}{2}M - \frac{1}{8}M^2 \right) + A_\perp N_2 \left(I_k - \frac{1}{2}M \right) \right] \cdot \\
&\left[B\left(I_k + \frac{1}{2}M^T - \frac{1}{8}\left(M^T\right)^2 \right) + B_\perp N_1 \left(I_k - \frac{1}{2}M^T \right) \right]^T .
\end{aligned}
$$

Proof The proof is given in Appendix C. ■

As a result from Lemma 4, we can calculate the retraction as the product of two low-rank factors: the first is an $n \times k$ matrix, the second a $k \times m$ matrix. Given a gradient $\tilde{\nabla}\mathcal{L}(x)$ in the ambient space, we can calculate the matrices M, N_1 and N_2 which allow us to represent its projection onto the tangent space, and furthermore allow us to calculate the retraction. We now have all the ingredients

Algorithm 2 : Naive Riemannian stochastic gradient descent

Input: Matrices $A \in \mathbb{R}_*^{n\times k}$, $B \in \mathbb{R}_*^{m\times k}$ s.t. $W = AB^T$. Gradient matrix $G \in \mathbb{R}^{n\times m}$ s.t. $G = -\eta\tilde{\nabla}\mathcal{L}(W) \in \mathbb{R}^{n\times m}$, where $\tilde{\nabla}\mathcal{L}(W)$ is the gradient in the ambient space and $\eta > 0$ is the step size.

Output: Matrices $Z_1 \in \mathbb{R}_*^{n\times k}$, $Z_2 \in \mathbb{R}_*^{m\times k}$ such that $Z_1 Z_2^T = R_W(-\eta\nabla\mathcal{L}(W))$.

Compute:	matrix dimension
$A^\dagger = (A^TA)^{-1}A^T$, $B^\dagger = (B^TB)^{-1}B^T$	$k\times n$, $k\times m$
$A_\perp$, $B_\perp$= orthogonal complements of A, B	$n\times(n-k)$, $m\times(m-k)$
$M = A^\dagger G B^{\dagger T}$	$k\times k$
$N_1 = B_\perp^T G^T A^{\dagger T}$	$(m-k)\times k$
$N_2 = A_\perp^T G B^{\dagger T}$	$(n-k)\times k$
$Z_1 = A\left(I_k + \frac{1}{2}M - \frac{1}{8}M^2\right) + A_\perp N_2\left(I_k - \frac{1}{2}M\right)$	$n\times k$
$Z_2 = B\left(I_k + \frac{1}{2}M^T - \frac{1}{8}(M^T)^2\right) + B_\perp N_1\left(I_k - \frac{1}{2}M^T\right)$	$m\times k$

necessary for a Riemannian stochastic gradient descent algorithm. The procedure is outlined in Algorithm 2.

Algorithm 2 explicitly computes and stores the orthogonal complement matrices $A_\perp$ and $B_\perp$, which in the low rank case $k \ll m,n$, have size $O(mn)$, the same as the full sized W. To improve the memory complexity, we use the fact that the matrices $A_\perp$ and $B_\perp$ always operate with their transpose. Since $A_\perp$ and $B_\perp$ have orthogonal columns, the matrix $A_\perp A_\perp^T$ is actually the projection matrix that we denoted earlier by $P_{A_\perp}$, and likewise for $B_\perp$. Because of orthogonal complementarity, these projection matrices are equal to $I_n - P_A$ and $I_m - P_B$ respectively. Thus we can write $A_\perp N_2 = \left(I - AA^\dagger\right)ZB^{\dagger T}$, and a similar identity for $B_\perp N_1$.

Consider now the case where the gradient matrix is of rank-r and is available in a factorized form $Z = G_1 G_2^T$, with $G_1 \in \mathbb{R}^{n\times r}$, $G_2 \in \mathbb{R}^{m\times r}$. Using the factorized gradient we can reformulate the algorithm to keep in memory only matrices of size at most $\max(n,m)\times k$ or $\max(n,m)\times r$. Optimizing the order of matrix operations so that the number of operations is minimized gives Algorithm 3. The runtime complexity of Algorithm 3 is readily computed based on matrix multiplications complexity,[2] and is $O\left((n+m)(k+r)^2\right)$.

3.2 LORETA With Rank-one Gradients

In many learning problems, the gradient matrix $\tilde{\nabla}\mathcal{L}(W)$ required for a gradient step update has a rank of one. This is the case for example, when the matrix model W acts as a bilinear form on two vectors, p and q, and the loss is a piecewise linear function of $\mathbf{p}^TW\mathbf{q}$ (as in Grangier and Bengio, 2008; Chechik et al., 2010; Weinberger and Saul, 2009; Shalev-Shwartz et al., 2004 and Section 7.1 below). In that case, the gradient is the rank-one outer product matrix $\mathbf{p}\mathbf{q}^T$. As another example, consider the case of multitask learning, where the matrix model W operates on a vector input $\mathbf{p}$, and the loss is the squared loss $\|W\mathbf{p} - \mathbf{q}\|^2$ between the multiple predictions $W\mathbf{p}$ and the true labels $\mathbf{q}$. The gradient of this loss is $(W\mathbf{p} - \mathbf{q})\mathbf{p}^T$, which is again a rank-one matrix. We now show how to

2. We assume throughout this paper the use of ordinary (schoolbook) matrix multiplication.

Algorithm 3 : **LORETA-r** - General-rank Riemannian stochastic gradient descent

Input: Matrices $A \in \mathbb{R}_*^{n\times k}$, $B \in \mathbb{R}_*^{m\times k}$ s.t. $W = AB^T$. Matrices $G_1 \in \mathbb{R}^{n\times r}$, $G_2 \in \mathbb{R}^{m\times r}$ s.t. $G_1 G_2^T = -\eta\tilde{\nabla}\mathcal{L}(W) \in \mathbb{R}^{n\times m}$, where $\tilde{\nabla}\mathcal{L}(W)$ is the gradient in the ambient space and $\eta > 0$ is the step size.

Output: Matrices $Z_1 \in \mathbb{R}_*^{n\times k}$, $Z_2 \in \mathbb{R}_*^{m\times k}$ such that $Z_1 Z_2^T = R_W(-\eta\nabla\mathcal{L}(W))$.

Compute:	matrix dimension	runtime complexity
$A^\dagger = (A^TA)^{-1}A^T, \quad B^\dagger = (B^TB)^{-1}B^T$	$k\times n, \quad k\times m$	$O((n+m)k^2)$
$\mathbf{a_1} = A^\dagger \cdot G_1, \quad \mathbf{b_1} = B^\dagger \cdot G_2$	$k\times r, \quad k\times r$	$O((n+m)kr)$
$\mathbf{a_2} = A\cdot\mathbf{a_1}$	$n\times r$	$O(nkr)$
$Q = \mathbf{b_1}^T\cdot\mathbf{a_1}$	$r\times r$	$O(kr^2)$
$\mathbf{a_3} = -\frac{1}{2}\mathbf{a_2} + \frac{3}{8}\mathbf{a_2}\cdot Q + G_1 - \frac{1}{2}G_1\cdot Q$	$n\times r$	$O(nr^2)$
$Z_1 = A + \mathbf{a_3}\cdot\mathbf{b_1}^T$	$n\times k$	$O(nkr)$
$\mathbf{b_2} = \left(G_2^TB\right)\cdot B^\dagger$	$r\times m$	$O(mkr)$
$\mathbf{b_3} = -\frac{1}{2}\mathbf{b_2} + \frac{3}{8}Q\cdot\mathbf{b_2} + G_2^T - \frac{1}{2}Q\cdot G_2^T$	$r\times m$	$O(mr^2)$
$Z_2^T = B^T + \mathbf{a_1}\cdot\mathbf{b_3}$	$k\times m$	$O(mkr)$

reduce the complexity of each iteration to be linear in the model rank k when the rank of the gradient matrix is $r = 1$.

Algorithm 4 : **LORETA-1** - Rank-one Riemannian stochastic gradient descent

Input: Matrices $A \in \mathbb{R}_*^{n\times k}$, $B \in \mathbb{R}_*^{m\times k}$ s.t. $W = AB^T$. Matrices $A^\dagger$ and $B^\dagger$, the pseudo-inverses of A and B respectively. Vectors $\mathbf{p} \in \mathbb{R}^{n\times 1}$, $\mathbf{q} \in \mathbb{R}^{m\times 1}$ s.t. $\mathbf{pq}^T = -\eta\tilde{\nabla}\mathcal{L}(W) \in \mathbb{R}^{n\times m}$, where $\tilde{\nabla}\mathcal{L}(W)$ is the gradient in the ambient space and $\eta > 0$ is the step size.

Output: Matrices $Z_1 \in \mathbb{R}_*^{n\times k}$, $Z_2 \in \mathbb{R}_*^{m\times k}$ s.t. $Z_1 Z_2^T = R_W(-\eta\nabla\mathcal{L}(W))$. Matrices $Z_1^\dagger$ and $Z_2^\dagger$, the pseudo-inverses of Z_1 and Z_2 respectively.

Compute:	matrix dimension	runtime complexity
$\mathbf{a_1} = A^\dagger\cdot\mathbf{p}, \mathbf{b_1} = B^\dagger\cdot\mathbf{q}$	$k\times 1$	$O((n+m)k)$
$\mathbf{a_2} = A\cdot\mathbf{a_1}$	$n\times 1$	$O(nk)$
$s = \mathbf{b_1}^T\cdot\mathbf{a_1}$	1×1	$O(k)$
$\mathbf{a_3} = \mathbf{a_2}\left(-\frac{1}{2}+\frac{3}{8}s\right) + \mathbf{p}(1-\frac{1}{2}s)$	$n\times 1$	$O(n)$
$Z_1 = A + \mathbf{a_3}\cdot\mathbf{b_1}^T$	$n\times k$	$O(nk)$
$\mathbf{b_2} = \left(\mathbf{q}^TB\right)\cdot B^\dagger$	$1\times m$	$O(mk)$
$\mathbf{b_3} = \mathbf{b_2}\left(-\frac{1}{2}+\frac{3}{8}s\right) + \mathbf{q}^T(1-\frac{1}{2}s)$	$1\times m$	$O(m)$
$Z_2^T = B^T + \mathbf{a_1}\cdot\mathbf{b_3}$	$k\times m$	$O(mk)$
$Z_1^\dagger = rank\_one\_pseudoinverse\_update(A, A^\dagger, \mathbf{a_3}, \mathbf{b_1})$	$k\times n$	$O(nk)$
$Z_2^\dagger = rank\_one\_pseudoinverse\_update(B, B^\dagger, \mathbf{b_3}, \mathbf{a_1})$	$k\times m$	$O(mk)$

Given rank-one gradients, the most computationally demanding step in Algorithm 3 is the computation of the pseudo-inverse of the matrices A and B, taking $O(nk^2)$ and $O(mk^2)$ operations. All other operations are $O(\max(n,m)\cdot k)$ at most. To speed up calculations we use the fact that for

$r = 1$ the outputs Z_1 and Z_2 become rank-one updates of the input matrices A and B. This enables us to keep the pseudo-inverses $A^\dagger$ and $B^\dagger$ from the previous round, and perform a rank-one update to them, following a procedure developed by Meyer (1973). The full procedure is included in Appendix D. This procedure is similar to the better known Sherman-Morrison formula for the inverse of a rank-one perturbed matrix, and its computational complexity for an $n \times k$ matrix is $O(nk)$ operations. Using that procedure, we derive our final algorithm, LORETA-*1*, the rank-one Riemannian stochastic gradient descent. Its overall time and space complexity are both $O((n+m)k)$ per gradient step. It can be seen that the LORETA-*1* algorithm uses only basic matrix operations, with the most expensive ones being low-rank matrix-vector multiplication and low-rank matrix-matrix addition. The memory requirement of LORETA-1 is about $4nk$ (assuming $m = n$), since it receives four input matrices of size nk ($A, B, A^\dagger, B^\dagger$) and assuming it can compute the four outputs ($Z_1, Z_2, Z_1^\dagger, Z_2^\dagger$), in-place while destroying previously computed terms.

4. Online Learning of Low-rank Positive Semidefinite Matrices

In this section we adapt the derivation above to the special case of positive semidefinite (PSD) matrices. PSD matrices are of special interest because they encode a true Euclidean metric. An n-by-n PSD matrix W of rank-k can be factored as $W = YY^T$, with $Y \in \mathbb{R}^{n\times k}$. Thus, the bilinear form $x^T W z$ is equal to $(Yx)^T(Yz)$, which is a Euclidean inner product in the space spanned by Y's columns. These properties have led to an extensive use of PSD matrix models in metric and similarity learning, see, for example, Xing et al. (2002), Goldberger et al. (2005), Globerson and Roweis (2006), Bar-Hillel et al. (2006) and Jain et al. (2008). The set of n-by-n PSD matrices of rank-k forms a manifold of dimension $nk - \frac{k(k-1)}{2}$, embedded in the Euclidean space $\mathbb{R}^{n\times n}$ (Vandereycken et al., 2009). We denote this manifold by $\mathcal{S}_+(k,n)$.

We now give a characterization of the tangent space of this manifold, due to Vandereycken and Vandewalle (2010).

Lemma 5 *Let $W \in \mathcal{S}_+(k,n)$ have a (non-unique) factorization $W = YY^T$, where $Y \in \mathbb{R}_*^{n\times k}$. Let $Y_\perp \in \mathbb{R}^{n\times(n-k)}$ be the orthogonal complement of Y such that $Y_\perp^T Y = 0$, $Y_\perp^T Y_\perp = I_{n-k}$. The tangent space to $\mathcal{S}_+(k,n)$ at W is:*

$$T_W\mathcal{S}_+(k,n) = \left\{ \begin{bmatrix} Y & Y_\perp \end{bmatrix} \begin{bmatrix} S & N^T \\ N & 0 \end{bmatrix} \begin{bmatrix} Y^T \\ Y_\perp^T \end{bmatrix} : S \in \mathbb{R}^{k\times k}, N \in \mathbb{R}^{(n-k)\times k}, S = S^T \right\}.$$

Proof See Vandereycken and Vandewalle (2010), Proposition 5.2. ■

Let $\xi \in T_W\mathcal{S}_+(k,n)$ be a tangent vector to $W = YY^T$. As shown by Vandereycken and Vandewalle (2010), ξ can be decomposed into two orthogonal components, $\xi = \xi^S + \xi^P$. Given a rank-r gradient matrix Z, and using the projection matrices P_Y and $P_{Y_\perp}$ they show that:

$$\xi^S = P_Y \frac{Z + Z^T}{2} P_Y,$$
$$\xi^P = P_{Y_\perp} \frac{Z + Z^T}{2} P_Y + P_Y \frac{Z + Z^T}{2} P_{Y_\perp}.$$

Using this characterization of the tangent vector when given an ambient gradient Z, one can define a retraction analogous to that defined in Section 3. This retraction is referred to as $R_W^{(2)}$ in Vandereycken and Vandewalle (2010).

Theorem 6 *Let $W \in \mathcal{S}_+(k,n)$, $W = YY^T$, and $W^\dagger$ be its pseudo-inverse. Let $\xi \in T_W\mathcal{S}_+(k,n)$, $\xi = \xi^S + \xi^P$, as described above, and let*

$$V = W + \frac{1}{2}\xi^S + \xi^P - \frac{1}{8}\xi^S W^\dagger \xi^S - \frac{1}{2}\xi^P W^\dagger \xi^S.$$

The mapping $R_W^{PSD}(\xi) = VW^\dagger V$ is a second order retraction from a neighborhood $\Theta_W \subset T_W\mathcal{S}_+(k,n)$ to $\mathcal{S}_+(k,n)$.

Proof See Vandereycken and Vandewalle (2010), Proposition 5.10. ■

Algorithm 5 : **LORETA-1-PSD** - Rank-one Riemannian PSD stochastic gradient descent

Input: A matrix $Y \in \mathbb{R}_*^{n\times k}$, s.t. $W = YY^T$. The matrix $Y^\dagger$, the pseudoinverse of Y. Vectors $\mathbf{p} \in \mathbb{R}^{n\times 1}$, $\mathbf{q} \in \mathbb{R}^{n\times 1}$ s.t. $pq^T = -\eta\tilde{\nabla}\mathcal{L}(W) \in \mathbb{R}^{n\times m}$, where $\tilde{\nabla}\mathcal{L}(W)$ is the gradient in the ambient space and $\eta > 0$ is the step size.

Output: Matrix $Z \in \mathbb{R}_*^{n\times k}$, s.t. $ZZ^T = R_W^{PSD}(-\eta\nabla\mathcal{L}(W))$. Matrix $Z^\dagger$, the pseudo-inverse of Z.

Compute:	matrix dimension	runtime complexity
$\mathbf{h_1} = Y^\dagger\mathbf{p}$	$k\times 1$	$O(nk)$
$\mathbf{h_2} = Y^\dagger\mathbf{q}$	$k\times 1$	$O(nk)$
$n_1 = \mathbf{h_1}^T\mathbf{h_1}$	1×1	$O(k)$
$n_2 = \mathbf{h_2}^T\mathbf{h_2}$	1×1	$O(k)$
$\hat{\mathbf{h_1}} = Y\mathbf{h_1}$	$n\times 1$	$O(nk)$
$\hat{\mathbf{h_2}} = Y\mathbf{h_2}$	$n\times 1$	$O(nk)$
$s = \mathbf{h_1}^T\mathbf{h_2}$	1×1	$O(k)$
$\mathbf{l_1} = (-\frac{1}{4} + \frac{3}{32}s)\hat{\mathbf{h_1}} + (\frac{1}{2} - \frac{1}{8}s)\mathbf{p} + \frac{3}{32}n_1\hat{\mathbf{h_2}} - \frac{1}{8}n_1\mathbf{q}$	$n\times 1$	$O(n)$
$\mathbf{l_2} = (-\frac{1}{4} + \frac{3}{32}s)\hat{\mathbf{h_2}} + (\frac{1}{2} - \frac{1}{8}s)\mathbf{q} + \frac{3}{32}n_2\hat{\mathbf{h_1}} - \frac{1}{8}n_2\mathbf{p}$	$n\times 1$	$O(n)$
$P_1 = \mathbf{l_1}\mathbf{h_2}^T$	$n\times k$	$O(nk)$
$P_2 = \mathbf{l_2}\mathbf{h_1}^T$	$n\times k$	$O(nk)$
$Z = Y + P_1 + P_2$	$n\times k$	$O(nk)$
$Z^\dagger_{temp} = rank\_one\_pseudoinverse\_update(Y, Y^\dagger, \mathbf{l_1}, \mathbf{h_2})$	$k\times n$	$O(nk)$
$Z^\dagger = rank\_one\_pseudoinverse\_update(Y + P_1, Z^\dagger_{temp}, \mathbf{l_2}, \mathbf{h_1})$	$k\times n$	$O(nk)$

Following the derivation of algorithms 2-4, and after some rearrangement, we obtain a PSD version of the LORETA-*1* algorithm. This PSD version is given in Algorithm (5). The algorithm is very similar to LORETA-*1* , but instead of learning a general rank-*k* matrix it learns a positive semidefinite rank-*k* matrix. The computational complexity and memory complexity of a gradient step for this algorithm is $O(nk)$, namely, it is linear in the reduced number of model parameters.

5. Manifold Identification

Until now, we formalized the problem of learning a low-rank matrix based on a factorization $W = AB$. At test time, computing the bilinear score using the model can be even faster when

the data is sparse. For instance, given two vectors x_1 and x_2 with c_1 and c_2 non-zero values, computing the bilinear form $x_1^T AB^T x_2$ requires $O(c_1k+k+kc_2) = O((c_1+c_2)k)$ operations, and can be significantly faster than the dense case. However, at training time, the LORETA-1 algorithm still has a complexity of $O((m+n)k)$ for each iteration even when the data is sparse.

The current section describes an attempt to adapt LORETA-1 such that it treats sparse data more efficiently. The empirical evaluation of this adaptation showed mixed results, but we include the derivation for completeness. The main idea is to separate the low-rank projection into two steps. First, a projection to a low dimensional space Ax that can be computed efficiently when x is sparse. Then, learning a second matrix, whose role is to tune the representation in the k-dimensional space.

To explain the idea, we focus on the case of learning a low-rank model which parametrizes a similarity function. The model is $W = AB^T$, $A \in \mathbb{R}^{n\times k}$, $B \in \mathbb{R}^{n\times k}$. The similarity between two vectors $\mathbf{p},\mathbf{q} \in \mathbb{R}^n$ is then given by

$$Sim(\mathbf{p},\mathbf{q}) = \mathbf{p}^T W \mathbf{q} = (A^T\mathbf{p})^T \cdot (B^T\mathbf{q}). \tag{3}$$

This similarity measure can be viewed as the cosine similarity in $\mathbb{R}^k$ between the projected vectors $B^T\mathbf{q}$ and $A^T\mathbf{p}$. We now introduce another similarity model which operates directly in the projected space. Formally, we have $M \in \mathbb{R}^{k\times k}$, and the similarity model is

$$Sim(\mathbf{p},\mathbf{q}) = (A^T\mathbf{p})^T M (B^T\mathbf{q}) = \mathbf{p}^T AMB^T\mathbf{q}. \tag{4}$$

Clearly, since the model in Equation (4) involves only linear matrix multiplications, its descriptive power is equivalent to that of the model Equation (3). However, it has the potential to be learned faster. To speed the training we can iterate between learning the outer projections A,B using LORETA , and learning the inner low-dimensional similarity model M using standard methods operating in the low-dimensional space. Specifically, the idea is to execute s update steps of M for every update step of A,B (Algorithm 6). After s update steps to M, it is decomposed using SVD to obtain $M = USV^T$, and these factors are used to update the outer projections using $A \leftarrow AUsqrt(S)$, $B \leftarrow BVsqrt(S)$.

Consider the computational complexity: Given two sparse vectors x_1, x_2 with c_1 and c_2 non-zero values respectively, projecting them using A and B to the low dimensional space takes $O(k(c_1+c_2))$, and an update step of M takes $O(k^2)$. Decomposing M using SVD takes $O(k^3)$, so the overall complexity for s updates is $O\left(k\cdot\left(s(k+c_1+c_2)+k^2\right)\right)$. When $s \geq k$ the cost of decomposition is amortized across many M updates and does not increase the overall complexity. The update of A, B takes $O(k(n+m))$ as before. This approach is related to the idea of manifold identification (Oberlin and Wright, 2007), where the learning of A, B "identifies" a manifold of rank k and the inner steps operate to tune the representation within that subspace.

This iterative procedure could be a significant speed up compared to the original $O((m+n)k)$. Unfortunately, when we tested this algorithm in a similarity learning task (as in Section 7.1), its performance was not as good as that of LORETA-1. The main reason was numerical instability: The matrix M typically collapsed to match few directions in A, and decomposing it has amplified the same A directions. This approach awaits deeper investigation which is outside the scope of the current paper.

6. Related Work

A recent summary of many advances in the field of optimization on manifolds is given by Absil et al. (2008). Advances in this field have lately been applied to matrix completion (Keshavan et al.,

Algorithm 6 : Manifold identification meta-algorithm

Input: Initial model matrices $A \in \mathbb{R}_*^{n\times k}$, $B \in \mathbb{R}_*^{m\times k}$ s.t. $W = AB^T$. Matrices $A^\dagger$ and $B^\dagger$, the pseudo-inverses of A and B respectively. Loss function $\mathcal{L}$.
Output: Matrices $A \in \mathbb{R}_*^{n\times k}$, $B \in \mathbb{R}_*^{m\times k}$ s.t. $W = AB^T$.
Parameters: η_1: LORETA step size . η_2: low-dimensional similarity learning step size. s: number of low-dimensional learning steps per round

repeat:
$[g_1, g_2] = \nabla\mathcal{L}(AB^T)$
$[A, B, A^\dagger, B^\dagger] = \text{LORETA}\ \left(A, B, A^\dagger, B^\dagger, g_1, g_2, \eta_1\right)$
initialize $M = I_k$
for i=1:s
$[g_1, g_2] = \nabla\mathcal{L}(AMB^T)$
$M = full - rank - metric - learning\left(M, A^T g_1, B^T g_2, \eta_2\right)$
endfor
$[U, S, V] = svd(M)$
$A = A \cdot U \cdot sqrt(S)$
$B = B \cdot V \cdot sqrt(S)$
until stopping condition is satisfied

2010), tensor-rank estimation (Eldén and Savas, 2009; Ishteva et al., 2011) and sparse PCA (Journée et al., 2010b).

Broadly speaking, there are two kinds of manifolds used in optimization. The first are *embedded manifolds*, manifolds that form a subset of Euclidean space, and are the ones we employ in this work. The second kind are *quotient manifolds*, which are formed by defining an equivalence relation on a Euclidean space, and endowing the resulting equivalence classes with an appropriate Riemannian metric. For example, the equivalence relation on $\mathbb{R}^n$ defined by $x \sim y \iff \exists\lambda > 0, x = \lambda y$, yields a quotient space called the *real projective space* when given a proper Riemannian metric.

More specific to the field of low-rank matrix manifolds, work has been done on the general problem of optimization with low-rank positive semi-definite (PSD) matrices. The latest and most relevant is the work of Meyer et al. (2011). In this work, Meyer and colleagues develop a framework for Riemannian stochastic gradient descent on the manifold of PSD matrices, and apply it to the problem of kernel learning and the learning of Mahalanobis distances. Their main technical tool is that of quotient manifolds mentioned above, as opposed to the embedded manifold we use in this work. Another paper which uses a quotient manifold representation is that of Journée et al. (2010a), which introduces a method for optimizing over low-rank PSD matrices.

In their 2010 paper (Vandereycken and Vandewalle, 2010), Vandereycken et al. introduced a retraction for PSD matrices in the context of modeling systems of partial differential equations. We build on this work in order to construct our methods of learning general and PSD low-rank matrices.

In general, the problem of minimizing a convex function over the set of low-rank matrices was addressed by several authors, including Fazel (2002). Recht et al. (2010) and more recently Jain et al. (2011) also consider the same problem, with additional affine constraints, and its connection to recent advances in compressed sensing. The main tools used in these papers are the trace norm

(sum of singular values) and semi-definite programming. See also Fazel et al. (2005) for a short introduction to these methods.

More closely related to the current paper are the papers by Kulis et al. (2009) and Meka et al. (2008). Kulis et al. (2009) deal with learning low-rank PSD matrices, and use the rank-preserving log-det divergence and clever factorization and optimization in order to derive an update rule with runtime complexity of $O(nk^2)$ for an $n \times n$ matrix of rank k. Meka et al. (2008) use online learning in order to find a minimal rank square matrix under approximate affine constraints. The algorithm does not directly allow a factorized representation, and depends on an "oracle" component, which typically requires to compute an SVD.

Multi-class ranking with a large number of features was studied by Bai et al. (2009), and in the context of factored representations, by Weston et al. (2011) (WSABIE). WSABIE combines projected gradient updates with a novel sampling scheme which is designed to minimize a ranking loss named WARP. WARP is shown to outperform simpler triplet sampling approaches. Since WARP yields rank-1 gradients, it can easily be adapted for Riemannian SGD, but we leave experiments with such sampling schemes to future work.

7. Experiments

We tested LORETA in two learning tasks: learning a similarity measure between pairs of text documents using the 20-newsgroups data collected by Lang (1995), and learning to rank image label annotations based on a multi-label annotated set, using the *ImageNet* data set (Deng et al., 2009). Matlab code for LORETA-1 is available online at *http://chechiklab.biu.ac.il/research/LORETA*.

7.1 Learning Similarity on the 20 Newsgroups Data Set

In our first set of experiments, we looked at the problem of learning a similarity measure between pairs of text documents. Similarity learning is a well studied problem, closely related to metric learning (see Yang 2007 for a review). It has numerous applications in information retrieval such as *query by example*, and finding related content on the web.

One approach to learn pairwise relations is to measure the similarity of two documents $\mathbf{p}, \mathbf{q} \in \mathbb{R}^n$ using a bilinear form parametrized by a model $W \in \mathbb{R}^{n \times n}$:

$$S_W(\mathbf{p}, \mathbf{q}) = \mathbf{p}^T W \mathbf{q}.$$

Such models can be learned online (Chechik et al., 2010) and were shown to achieve high precision. Sometimes the matrix W is required to be symmetric and positive definite, which means it actually encodes a metric, also known as a Mahalanobis distance. Unfortunately, since the number of parameters grows as n^2, storing the matrix W in memory is only feasible for limited feature dimensionality. To handle larger vocabularies, like those containing all textual terms found in a corpus, a common approach is to pre-select a subset of the features and train a model over the low dimensional data. However, such preprocessing may remove crucial signals in the data even if features are selected in a discriminative way.

To overcome this difficulty, we used LORETA-1 and LORETA-1-PSD to learn a rank-k parametrization of the model W. This model can be factorized as $W = AB^T$, where $A, B \in \mathbb{R}^{n \times k}$ for the general case, or as $W = AA^T$ for the PSD case. In each of our experiments, we selected a subset of n features, and trained a rank k model. We varied the number of features n and the rank of the matrix k

so as to use a fixed amount of memory. For example, we used a rank-10 model with $50K$ features, and a rank-50 model with $10K$ features.

7.1.1 SIMILARITY LEARNING WITH LORETA-1

We use an online procedure similar to that in Grangier and Bengio (2008) and Chechik et al. (2010). At each round, three instances are sampled: a query document $\mathbf{q} \in \mathbb{R}^n$, and two documents $\mathbf{p}_+, \mathbf{p}_- \in \mathbb{R}^n$ such that $\mathbf{p}_+$ is known to be more similar to $\mathbf{q}$ than $\mathbf{p}_-$. We wish that the model assigns a higher similarity score to the pair $(\mathbf{q}, \mathbf{p}_+)$ than the pair $(\mathbf{q}, \mathbf{p}_-)$, and hence use the online ranking hinge loss defined as $l_W(\mathbf{q}, \mathbf{p}_+, \mathbf{p}_-) = [1 - S_W(\mathbf{q}, \mathbf{p}_+) + S_W(\mathbf{q}, \mathbf{p}_-)]_+$, where $[z]_+ = max(z, 0)$.

We initialized the model to be a truncated identity matrix, with only the first k ones along the diagonal. This corresponds in our case to choosing the k most informative terms as the initial data projection.

7.1.2 DATA PREPROCESSING AND FEATURE SELECTION

We used the 20 newsgroups data set (people.csail.mit.edu/jrennie/20Newsgroups), containing 20 classes with approximately 1000 documents each. We removed stop words but did not apply stemming. The document terms form a vocabulary of 50,000 terms, and we selected a subset of these features that conveyed high information about the identity of the class (over the training set) using the *infogain* criterion (Yang and Pedersen, 1997). This is a discriminative criterion,which measures the number of bits gained for category prediction by knowing the presence or absence of a term in a document. The selected features were normalized using *tf-idf*, and then represented each document as a bag of words. Two documents were considered similar if they shared the same class label, out of the possible 20 labels.

7.1.3 EXPERIMENTAL PROCEDURE AND EVALUATION PROTOCOL

The 20 newsgroups site proposes a split of the data into train and test sets. We repeated splitting 5 times based on the sizes of the proposed splits (a train / test ratio of 65% / 35%). We evaluated the learned similarity measures using a ranking criterion. We view every document $\mathbf{q}$ in the test set as a query, and rank the remaining test documents $\mathbf{p}$ by their similarity scores $\mathbf{q}^T W \mathbf{p}$. We then compute the precision (fraction of positives) at the top r ranked documents. We then average the precision over all positions r such that there exists a positive example in the top r. This final measure is called *mean average precision*, and is commonly used in the information retrieval community (Manning et al., 2008, Chapter 8).

7.1.4 COMPARISONS

We compared LORETA with the following approaches.

1. **Naive gradient descent** (GD): similar to Bai et al. (2009). The model is represented as a product of two matrices $W = AB^T$. Stochastic gradient descent steps are computed over the factors A and B, for the same loss used by LORETA $l_W(\mathbf{q}, \mathbf{p}_+, \mathbf{p}_-)$. The GD steps are:

$$A_{new} = A - \eta\, \mathbf{q}(\mathbf{p}_- - \mathbf{p}_+)^T B,$$
$$B_{new} = B - \eta\, (\mathbf{p}_- - \mathbf{p}_+)\mathbf{q}^T A.$$

   We found this approach to be very unstable, and thus its results are not presented.

2. **Naive PSD gradient descent**: similar to the method above, except that now the model is constrained to be PSD. The model is represented as a product $W = AA^T$. Stochastic gradient descent steps are computed over the factor A for the same loss used by LORETA : $l_W(\mathbf{q}, \mathbf{p}_+, \mathbf{p}_-)$. As shown by Meyer et al. (2011), this is in fact equivalent to Riemannian stochastic GD in the manifold of PSD matrices when this manifold is endowed with a certain metric the authors call the *flat metric*.

   The GD step is:

$$A_{new} = A - \eta \left(\mathbf{q}(\mathbf{p}_- - \mathbf{p}_+)^T + (\mathbf{p}_- - \mathbf{p}_+)\mathbf{q}^T\right) A.$$

   The step size η was chosen by cross validation. This approach was more stable in the PSD case than in the general case, probably because the invariant space here is only the group of orthogonal matrices (which are well-conditioned), as opposed to the group of invertible matrices which might be ill-conditioned.

3. **Iterative Passive-Aggressive (PA)**: since we found the above general GD procedure **(1)** to be very unstable, we experimented with a related online algorithm from the family of passive-aggressive algorithms (Crammer et al., 2006). We iteratively optimize over A given a fixed B and vice versa. The optimization is a tradeoff between minimizing the loss l_W, and limiting how much the models change at each iteration. The steps sizes for updating A and B are computed to be:

$$\begin{aligned}
\eta_A &= \min\left(\frac{l_W(\mathbf{q}, \mathbf{p}_+, \mathbf{p}_-)}{\|\mathbf{q}\|^2 \cdot \|B^T(\mathbf{p}_+ - \mathbf{p}_-)\|^2}, C\right), \\
\eta_B &= \min\left(\frac{l_W(\mathbf{q}, \mathbf{p}_+, \mathbf{p}_-)}{\|(\mathbf{p}_+ - \mathbf{p}_-)\|^2 \cdot \|A^T\mathbf{q}\|^2}, C\right).
\end{aligned}$$

   C is a predefined parameter controlling the maximum magnitude of the step size, chosen by cross-validation. This procedure is numerically more stable because of the normalization by the norms of the matrices multiplied by the gradient factors.

4. **Full rank similarity learning models.** We compared with two full rank online metric learning methods, LEGO (Jain et al., 2008) and OASIS (Chechik et al., 2010). Both algorithms learn a full (non-factorized) model, and were run with $n = 1000$, in order to be consistent with the memory constraint of LORETA-1. We have also compared with both full-rank models using rank 2000, that is, 4 times the memory constraint. We have not compared with batch approaches such as Kulis et al. (2009), since they are not expected to scale to very large data sets such as those our work is ultimately aiming towards.

In addition, we have experimented with the method for learning PSD matrices using a polar geometry characterization of the quotient manifold, due to Meyer et al. (2011). This method's runtime complexity is $O((n+m)k^2)$, and we have found that its performance was not in line with the methods described above.

7.1.5 RESULTS

Figure 3c shows the mean average precision obtained with all the above methods. LORETA outperforms the other approaches across all ranks. LORETA-PSD achieves slightly higher precision

than LORETA. The reason may be that similarity was defined based on two samples belonging to a common class, and this relation is symmetric and transitive, two relations which are respected by PSD matrices but not by general similarity matrices. Moreover, LORETA-PSD learned faster along the training iterations when compared with LORETA - see Figure 3a for a comparison of the learning curves. Interestingly, for both LORETA algorithms learning a low-rank model of rank 30, using the best 16660 features, was significantly more precise than learning a much fuller model of rank 100 and 5000 features, or a model using the full 50000 word vocabulary but with rank 10 . The intuition is that LORETA can be viewed as adaptively learning a linear projection of the data into low dimensional space, which is tailored to the pairwise similarity task.

7.2 Image Multilabel Ranking

Our second set of experiments tackled the problem of learning to rank labels for images taken from a large number of classes ($L = 1660$) with multiple labels per image.

In our approach, we learn a linear classifier over n features for each label $c \in \mathcal{C} = \{1,\ldots,L\}$, and stack all models together to a single matrix $W \in \mathbb{R}^{L\times n}$. At test time, given an image $\mathbf{p} \in \mathbb{R}^n$, the product $W\mathbf{p}$ provides scores for every label for that image $\mathbf{p}$. Given ground truth labeling, a good model would rank the true labels higher than the false ones. Each row of the matrix model can be thought of as a sub-model for the corresponding label. Imposing a low-rank constraint on the model implies that these sub-models are linear combinations of a smaller number of latent models. Alternatively, we can view learning a factored rank-k model $W = AB^T$ as learning a projection and classifier in the projected space concurrently. The matrix B^T projects the data onto a k dimensional space, and the matrix A consists of L classifiers operating in the low-dimensional space. The data we used for the experiment had $\sim$1500 labels, but the full ImageNet data set currently has $\sim$15000 labels, and is growing.

7.2.1 ONLINE LEARNING OF LABEL RANKINGS WITH LORETA-1

At each iteration, an image $\mathbf{p}$ is sampled, and using the current model W the scores for all its labels are computed, $W\mathbf{p}$. These scores are compared with the ground truth labeling $\mathbf{y} = \{y_1,\ldots,y_r\} \subset \mathcal{C}$. We wish for all the scores of the true labels to be higher than the scores for the other labels by a margin of 1. Thus, the learner suffers a multilabel multiclass hinge loss as follows. Let $\bar{y} = \text{argmax}_{s\notin\mathbf{y}}(W\mathbf{p})_s$, be the negative label which obtained the highest score, where $(W\mathbf{p})_s$ is the s^{th} component of the score vector $W\mathbf{p}$.

The loss is then $\mathcal{L}(W,\mathbf{p},\mathbf{y}) = \sum_{i=1}^{r}\left[(W\mathbf{p})_{\bar{y}} - (W\mathbf{p})_{y_i} + 1\right]_+$, which is the sum of the margins between the top-ranked false label and all the positive labels which violated the margin of one from it. We used the subgradient G of this loss for LORETA: for the set of indices $i_1, i_2, \ldots i_d \subset \mathbf{y}$ which incurred a non zero hinge loss, the i_j row of G is $\mathbf{p}$, and for the row $\bar{y}$ G is $-d\cdot\mathbf{p}$. The matrix G is rank one, unless no loss was suffered in which case it is 0.

The non-convex and stochastic nature of the learning procedure has lead us to try several initial conditions:

- **Zero matrix**: in this initialization we begin with a low-rank matrix composed entirely of zeros. This matrix is not included in the low-rank manifold $\mathcal{M}_k^{n,m}$, since its rank is less than k. We therefore perform a simple pre-training session in which we add up subgradients until a matrix of rank k is obtained. In practice we added the first $2k$ subgradients (each such subgradient being of rank one), and then performed an SVD to obtain the best rank-k model.

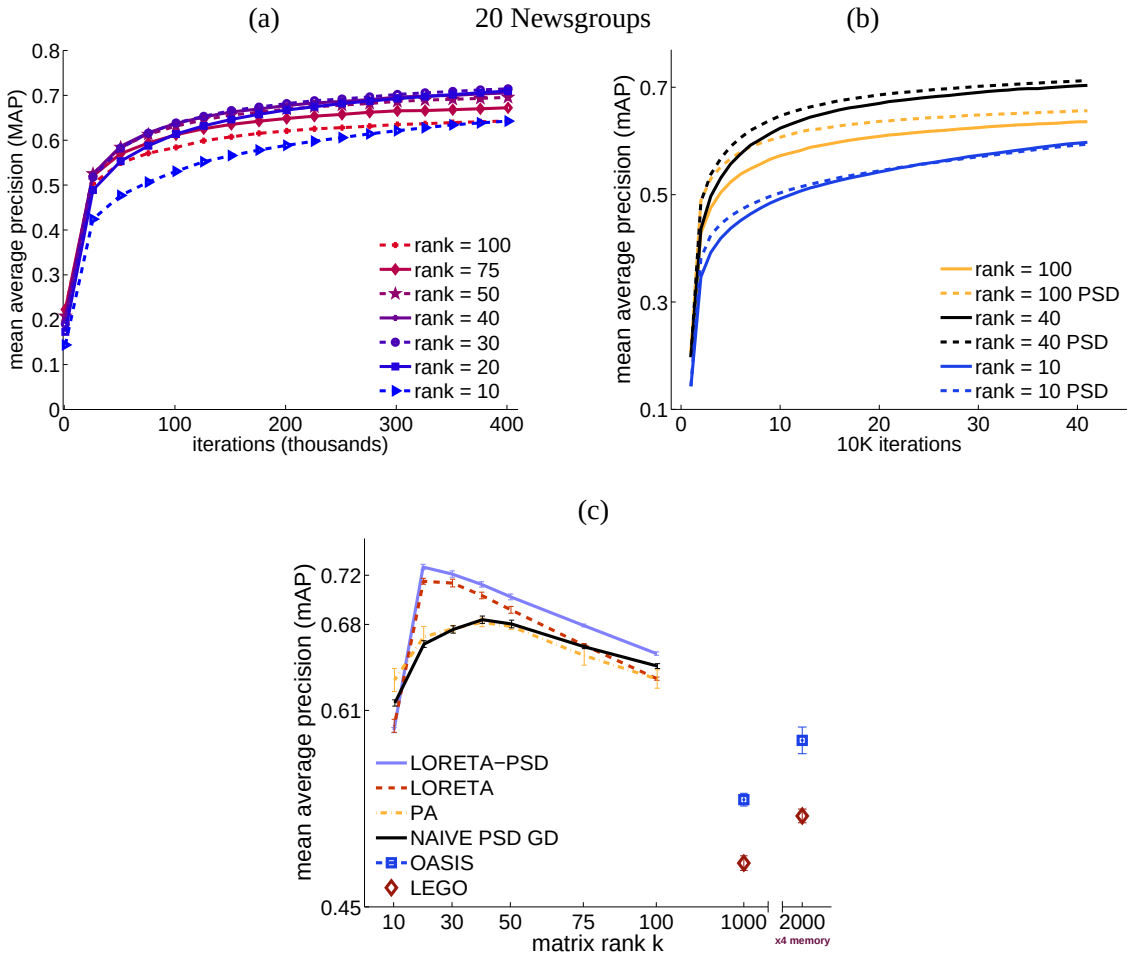


Figure 3: (a) Mean average precision (mAP) over 20 newsgroups test set as traced along LORETA learning for various ranks. Curve values are averages over 5 train-test splits. (b) Comparison of the learning curves of LORETA and LORETA-PSD. LORETA-PSD learns faster than LORETA across all ranks (shown are results for ranks 10, 40 and 100). (c) mAP of different models with varying rank. For each rank, a different number of features was selected using an information gain criterion, such that the total memory requirement is kept fixed (number of features $\times$ rank is constant). 50000 features were used for rank = 10. LEGO and OASIS were trained with the same memory (using 1000 features and rank=1000), as well as with 4 times the same memory (rank=2000). Error bars denote the standard error of the mean over 5 train-test splits.

We chose $2k$ because we wanted to ensure that the matrix we obtain has rank greater or equal to k.

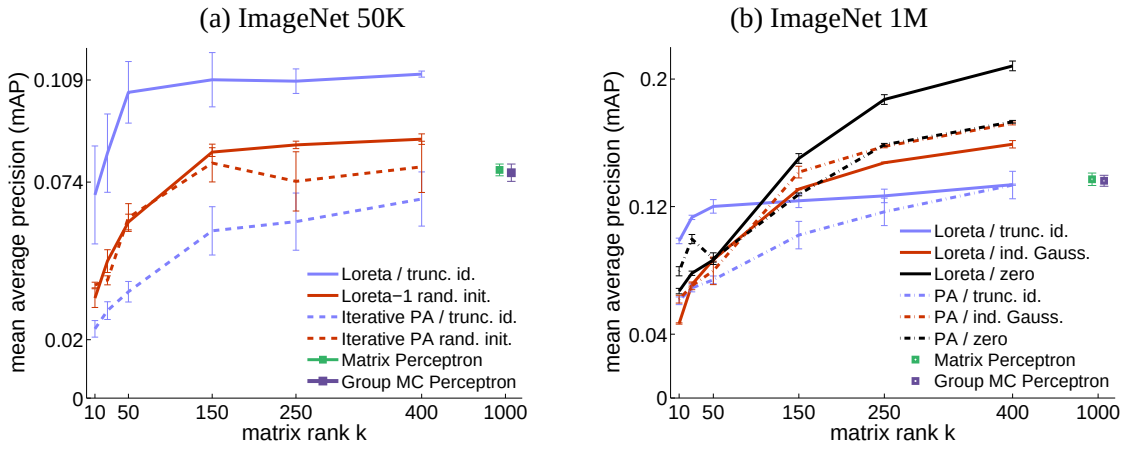


Figure 4: ImageNet data. Mean average precision (mAP) as a function of the rank k. Curves are means over five train-test splits. Error bars denote the standard error of the mean. Note the different scale of the left and right figure. All hyper parameters were selected using cross validation. Three different initializations were used: the zero matrix, a zero padded $k \times k$ identity matrix, and a product of two i.i.d. Gaussian matrices. See Section 7.2.1 for details.

- **Zero-padded identity**: we begin with a matrix composed of the $k \times k$ identity matrix I_k on the top left corner, padded with zeros so as to form an $L \times n$ matrix. This is guaranteed to be of rank k. The choice of the location of the identity matrix block is arbitrary.

- **Independent Gaussian**: we sample independently the entries of the two factor matrices $A \in \mathbb{R}^{n \times k}$, $B\mathbb{R}^{m \times k}$ from a standard normal distribution. This model is thus initialized as a product of two random Gaussian matrices.

7.2.2 DATA SET AND PREPROCESSING

We used data from the ImageNet 2010 Challenge (www.imagenet.org/challenges/LSVRC/2010/) containing images labeled with respect to the WordNet hierarchy. Each image was manually labeled with a single class label (for a total of 1000 classes). We added labels for each image, using classes along the path to the root of the hierarchy (adding 676 classes in total). We discarded ancestor labels covering more than 10% of the images, leaving 1660 labels (5.2 labels per image on average). We used ImageNet's bag of words representation, based on vector quantizing SIFT features with a vocabulary of 1000 words, followed by *tf-idf* normalization.

7.2.3 EXPERIMENTAL PROCEDURE AND EVALUATION PROTOCOL

We trained on two data sets. A medium scale one of 50000 images, and a large data set consisting of 908210 images. We tested on 20000 images for the medium scale, and 252284 images for the large scale. The quality of the learned label ranking was evaluated using the *mean average precision* (mAP) criterion mentioned in 7.1.3 above (Manning et al., 2008, Chapter 8).

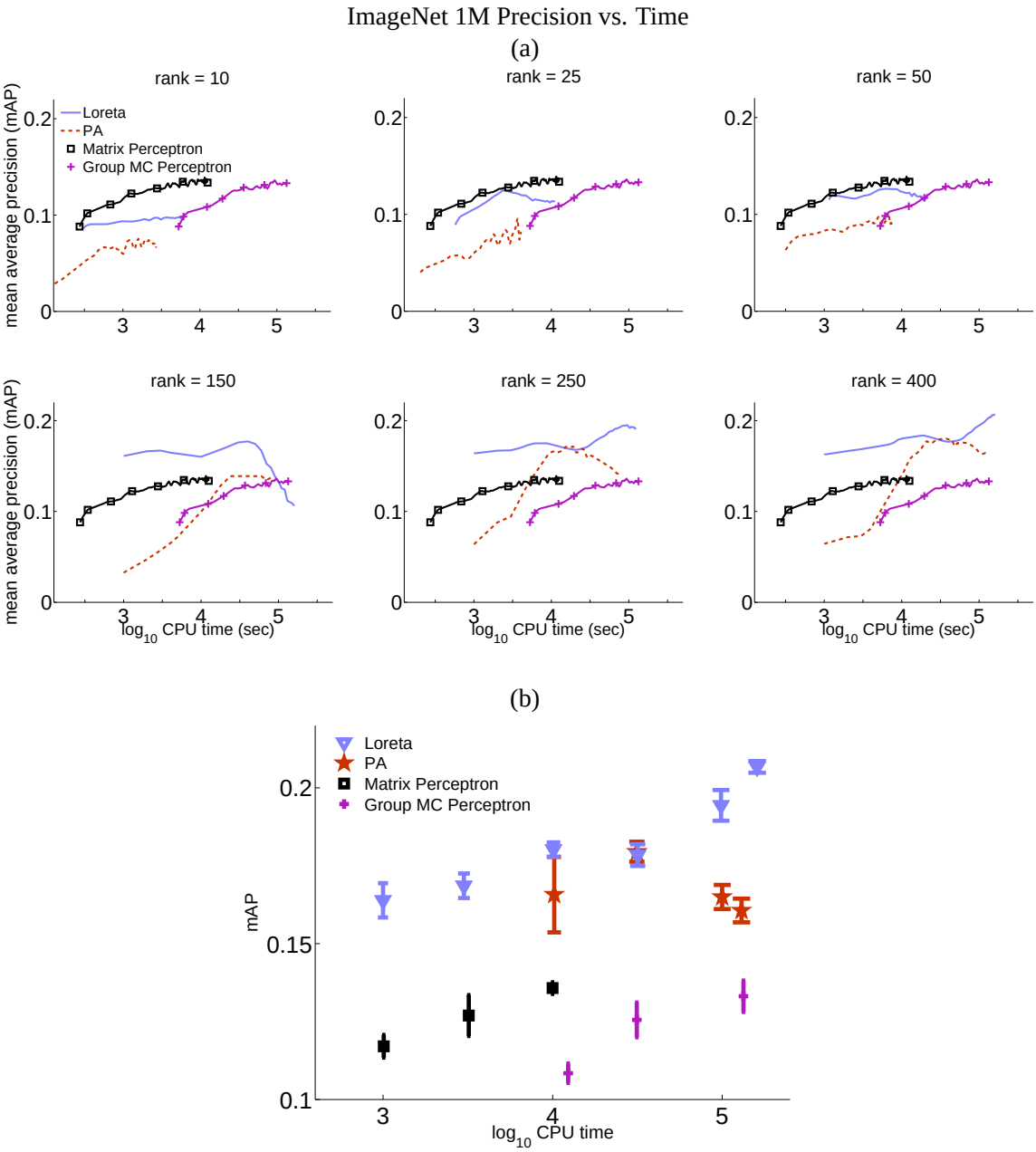


Figure 5: (a) Mean average precision (mAP) as function of single CPU processing time in seconds for different algorithms and model ranks, presented on a log-scale. Matrix Perceptron (black squares) and Group Multi-Class Perceptron (purple crosses) are both full rank (rank=1000), and their curves are reproduced on all six panels for comparison. For each rank and algorithm (LORETA and PA), we used the best performing initialization scheme. (b) mAP of the best performing model for different algorithms and time points. Error bars represent standard deviation across 5 train-test splits.

7.2.4 COMPARISONS

We compared the performance of LORETA on this task with three other approaches:

1. **PA - Iterative Passive-Aggressive**: same as described in Section 7.1.4 above for the 20 Newsgroups experiment.

2. **Matrix Perceptron**: a full rank stochastic subgradient descent. The model is initialized as a zero matrix of size 1660×1000, and in each round the loss subgradient is subtracted from it. After a sufficient number of rounds, the model is typically full rank and dense.

3. **Group Multi-Class Perceptron**: a mixed (2,1) norm online mirror descent algorithm (Kakade et al., 2010). This algorithm encourages a group-sparsity pattern within the learned matrix model, thus presenting an alternative form of regularization when compared with low-rank models.

LORETA and PA were run using a range of model ranks. For all three methods, the step size (or the C parameter for PA) was chosen by 5-fold cross validation on a validation set.

7.2.5 RESULTS

Figure 4 plots the mAP precision of LORETA and PA for different model ranks, while showing on the right the mAP of the full rank 1000 Matrix Perceptron and $(2,1)$ norm algorithms. LORETA significantly improves over all other methods across all ranks. However, we note that LORETA, being a non-convex algorithm, does depend significantly on the method of initialization, with the zero-padded identity matrix being the best initialization for lower rank models, and the zero matrix the best initialization for higher rank models (rank ≥ 150).

In Figure 5 we show the accuracy as a function of CPU tim on a single CPU for the different algorithms and model ranks. We ran Matlab R2011a on an Intel Xeon 2.27 GHz machine, and used Matlab's `-singlethread` flag to control multithreading. The higher-rank LORETA models outperform all others both in the short time scale (~ 1000 sec.) and the long time scale ($\sim 100,000$ sec.). For some of the higher-rank models there is evident overtraining at some point, but this overtraining could be avoided by adopting an early-stopping procedure.

8. Discussion

We presented LORETA, an algorithm which learns a low-rank matrix based on stochastic Riemannian gradient descent and efficient retraction to the manifold of low-rank matrices. LORETA achieves superior precision in a task of learning similarity in high dimensional feature spaces, and in multi-label annotation, where it scales well with the number of classes. A PSD variant of LORETA can be used efficiently for low-rank metric learning.

There are many ways to tie together different classifiers in a multi-class setting. We have seen here that a low-rank assumption coupled with a Riemannian SGD procedure outperformed the (2,1) mixed norm. Other approaches leverage the hierarchical structure inherent in many of these tasks. For example, Deng et al. (2011) use the label hierarchy of ImageNet to compute a similarity measure between images.

For similarity learning, the approach we take in this paper uses a weak supervision based on ranking similar pairs: one only knows that the pair $(\mathbf{q}, \mathbf{p}_+)$ is more similar than the pair $(\mathbf{q}, \mathbf{p}_-)$. In

some cases, a stronger supervision signal is available, like the classes of each objects are known. In these cases, Deng et al. (2011) have shown how to use class identities to construct good features by training an SVM classifier on each class and using its scaled output as a feature. They show that such features can lead to very good performance, with the added advantage that the features can be learned in parallel. The weak supervision approach that we take here aims to handle the case, which is particularly common in large scale data sets collected through web users' activity, where weaker supervision is much easier to collect.

In this paper, we used simple sampling schemes for both the similarity learning and multiple-labelling experiments. More elaborate sampling techniques such as those proposed by Weston et al. (2011), which focus on "hard negatives", may yield significant performance improvements. As these approaches typically involve rank-one gradients when implemented as online learning algorithms, they are well suited for being used in conjunction with LORETA, and this will be the subject of future work.

LORETA yields a factorized representation of the low-rank matrix. For similarity learning, these factors project to a low-dimensional space where similarity is evaluated efficiently. For classification, it can be viewed as learning two matrix components: one that projects the high dimensional data into a low dimension, and a second that learns to classify in the low dimension. In both approaches, the low-dimensional space is useful for extracting the relevant structure from the high-dimensional data, and for exploring the relations between large numbers of classes.

Acknowledgments

G.C. was supported by the Israeli science foundation Grant 1001/08, and by a Marie Curie reintegration grant PIRG06-GA-2009-256566. U.S. and D.W. were supported by the European Union under the DIRAC integrated project IST-027787. Uri Shalit is a recipient of the Google Europe Fellowship in Machine Learning, and this research is supported in part by this Google Fellowship. We would also like to thank the Gatsby Charitable Foundation for its generous support of the ICNC.

Appendix A. Proof of Lemma 2

We formally define the tangent space of a manifold at a point on the manifold, and then describe an auxiliary parametrization of the tangent space to the manifold $\mathcal{M}_k^{n,m}$ at a point $W \in \mathcal{M}_k^{n,m}$.

Definition 7 *The tangent space $T_W\mathcal{M}$ to a manifold $\mathcal{M} \subset \mathbb{R}^n$ at a point $W \in \mathcal{M}$ is the linear space spanned by all the tangent vectors at 0 to smooth curves $\gamma : \mathbb{R} \to \mathcal{M}$ such that $\gamma(0) = W$. That is, the set of tangents in $\mathbb{R}^n$ to smooth curves within the manifold which pass through the point W.*

In order to characterize the tangent space of $\mathcal{M}_k^{n,m}$, we look into the properties of smooth curves γ, where for each t, $\gamma(t) \in \mathcal{M}_k^{n,m}$.

For any such curve, because of the rank k assumption, we may assume that for all $t \in \mathbb{R}$, there exist (non-unique) matrices $A(t) \in \mathbb{R}_*^{n\times k}$, $B(t) \in \mathbb{R}_*^{m\times k}$, such that $\gamma(t) = A(t)B(t)^T$. We now wish to find the tangent vectors to these curves. By the product rule we have:

$$\dot{\gamma}(0) = A(0)\dot{B}(0)^T + \dot{A}(0)B(0)^T.$$

Since $W = \gamma(0) = A(0)B(0)^T = AB^T$ we have for $W = AB^T$:

$$T_{\mathbf{W}}\mathcal{M}_k^{n,m} = \left\{ AX^T + YB^T \,|\, X \in \mathbb{R}^{m\times k}, Y \in \mathbb{R}^{n\times k} \right\}. \tag{5}$$

This is because any choice of matrices X, Y such that $X = \dot{B}, Y = \dot{A}$ will give us some tangent vector, and for any tangent vector there exist such matrices. The space above is clearly a linear space. Being a tangent space to a manifold, it has the same dimension as the manifold: $(n+m)k - k^2$.

Recall the definition of the tangent space given in Lemma 1:

$$T_{\mathbf{W}}\mathcal{M}_k^{n,m} = \left\{ \begin{bmatrix} A & A_\perp \end{bmatrix} \begin{bmatrix} M & N_1^T \\ N_2 & 0 \end{bmatrix} \begin{bmatrix} B^T \\ B_\perp^T \end{bmatrix} : M \in \mathbb{R}^{k\times k}, N_1 \in \mathbb{R}^{(m-k)\times k}, N_2 \in \mathbb{R}^{(n-k)\times k} \right\}. \tag{6}$$

To prove Lemma 2, it is easy to verify by counting that the dimension of the space as defined in Equation (6) above is $(n+m)k - k^2$. Using the notation above, we can see that by taking $X = MB^T + N_1 B_\perp^T$ and $Y = A_\perp N_2$, the space defined in Equation (6) is included in $T_{\mathbf{W}}\mathcal{M}_k^{n,m}$ as defined in Equation (5). Since it is a linear subspace of equal dimension, both spaces must be equal ■

Appendix B. Proof of Theorem 3

We state the theorem again here.

Theorem 8 *Let $W \in \mathcal{M}_k^{n,m}$, $W = AB^T$, and $W^\dagger = B^{\dagger T}A^\dagger$. Let $\xi \in T_W\mathcal{M}_k^{n,m}$, $\xi = \xi^{AB} + \xi^{AB_\perp} + \xi^{A_\perp B}$, as in 1, and let:*

$$V_1 = W + \frac{1}{2}\xi^{AB} + \xi^{A_\perp B} - \frac{1}{8}\xi^{AB}W^\dagger\xi^{AB} - \frac{1}{2}\xi^{A_\perp B}W^\dagger\xi^{AB} \quad ,$$
$$V_2 = W + \frac{1}{2}\xi^{AB} + \xi^{AB_\perp} - \frac{1}{8}\xi^{AB}W^\dagger\xi^{AB} - \frac{1}{2}\xi^{AB}W^\dagger\xi^{AB_\perp} \quad .$$

The mapping

$$R_W(\xi) = V_1 W^\dagger V_2 \tag{7}$$

is a second order retraction from a neighborhood $\Theta_W \subset T_W\mathcal{M}_k^{n,m}$ to $\mathcal{M}_k^{n,m}$.

Proof To prove that Equation (7) defines a retraction, we first show that $V_1W^\dagger V_2$ is a rank-k matrix. Note that there exist matrices $Z_1 \in \mathbb{R}^{n\times k}$ and $Z_2 \in \mathbb{R}^{m\times k}$ such that $V_1 = Z_1B^T$ and , $V_2 = AZ_2^T$. A sufficient condition for the matrices Z_1 and Z_2 to be of full rank is that the matrix M is of limited norm. Thus, for all tangent vectors lying in some neighborhood $\Theta_W \subset T_W\mathcal{M}_k^{n,m}$ of $0 \in T_W\mathcal{M}_k^{n,m}$, the above relation is indeed a retraction to the manifold. In practice this is never a problem, as the set of matrices not of full rank is of zero measure, and in practice we have found these matrices to always be of full rank. Thus, $R_W(\xi) = V_1W^\dagger V_2 = Z_1B^TB(B^TB)^{-1}(A^TA)^{-1}A^TAZ_2^T = Z_1Z_2^T$, which, given that Z_1 and Z_2 are of full column rank, is exactly a rank-k, $n \times m$ matrix.

Next we show that $R_W(\xi)$ is a retraction, and of second order. It is obvious that $R_W(0) = W$, since the projection of the zero vector is zero, and thus ξ^{AB}, $\xi^{AB_\perp}$ and $\xi^{A_\perp B}$ are all zero.

Expanding $V_1W^\dagger V_2$ up to second order terms in ξ, many terms cancel and we end up with:

$$\begin{aligned} R_W(\xi) &= W + \xi^{AB} + \xi^{AB_\perp} + \xi^{A_\perp B} + \xi^{A_\perp B}W^\dagger\xi^{AB_\perp} + O(\|\xi\|^3) \\ &= W + \xi + \xi^{A_\perp B}W^\dagger\xi^{AB_\perp} + O(\|\xi\|^3). \end{aligned}$$

Local first order rigidity is immediately apparent. If we expand the only second order term, $\xi^{A_\perp B}W^\dagger\xi^{AB_\perp}$, we see that it equals $A_\perp N_2 N_1^T B_\perp^T$. We claim this term is orthogonal to the tangent space $T_W\mathcal{M}_k^{n,m}$. If we take, using the characterization in Lemma 2, an arbitrary tangent vector $A\tilde{M}B^T + A\tilde{N}_1^T B_\perp^T + A_\perp\tilde{N}_2 B^T$ in $T_{\mathbf{W}}\mathcal{M}_k^{n,m}$, we can calculate the inner product:

$$\begin{aligned}
&\left\langle \left(A_\perp N_2 N_1^T B_\perp^T\right), \left(A\tilde{M}B^T + A\tilde{N}_1^T B_\perp^T + A_\perp\tilde{N}_2 B^T\right)\right\rangle = \\
&tr\left(B_\perp N_1 N_2^T A_\perp^T A\tilde{M}B^T + B_\perp N_1 N_2^T A_\perp^T A\tilde{N}_1^T B_\perp^T + B_\perp N_1 N_2^T A_\perp^T A_\perp\tilde{N}_2 B^T\right) = \\
&tr\left(B_\perp N_1 N_2^T A_\perp^T A_\perp\tilde{N}_2 B^T\right) = \\
&tr\left(B^T B_\perp N_1 N_2^T A_\perp^T A_\perp\tilde{N}_2\right) = 0
\end{aligned}$$

with the equalities stemming from the fact that $A_\perp^T A = 0$, $B_\perp^T B = 0$, and from standard trace identities. Thus, the second order term cancels out if we project the second derivative of the curve defined by the retraction, as required by the second-order condition

$$P_W\left(\frac{\mathrm{d}R_W(\tau\xi)}{\mathrm{d}\tau^2}|_{\tau=0}\right) = 0 \quad \forall\xi \in T_W\mathcal{M}.$$

We see that the second order term is contained in the normal space. This concludes the proof that the retraction is a second order retraction. ■

Appendix C. Proof of Lemma 4

Let us see how can we calculate the needed terms explicitly. When evaluating the expression $V_1W^\dagger V_2$, we can use the algebraic relations: $WW^\dagger = P_A$ and $W^\dagger W = P_B$. From this we can conclude that: $WW^\dagger\xi^{AB} = \xi^{AB}$, $\xi^{AB}W^\dagger W = \xi^{AB}$, $\xi^{A_\perp B}W^\dagger W = \xi^{A_\perp B}$ and $WW^\dagger\xi^{AB_\perp} = \xi^{AB_\perp}$. These relations, along with many terms that cancel out, lead to the following expression:

$$\begin{aligned}
R_W(\xi) = V_1W^\dagger V_2 = \\
&W + \xi^{AB} + \xi^{AB_\perp} + \xi^{A_\perp B} - \frac{1}{8}\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB} - \frac{3}{8}\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB_\perp} \\
&- \frac{3}{8}\xi^{A_\perp B}W^\dagger\xi^{AB}W^\dagger\xi^{AB} + \xi^{A_\perp B}W^\dagger\xi^{AB_\perp} - \xi^{A_\perp B}W^\dagger\xi^{AB}W^\dagger\xi^{AB_\perp} \\
&+ \frac{1}{16}\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB_\perp} + \frac{1}{16}\xi^{A_\perp B}W^\dagger\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB} \\
&+ \frac{1}{64}\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB} + \frac{1}{4}\xi^{A_\perp B}W^\dagger\xi^{AB}W^\dagger\xi^{AB}\xi^{AB_\perp}.
\end{aligned}$$

We now substitute the matrices M, N_1 and N_2 into the above relation. Most terms cancel out. For example, we have the identity $\xi^{AB}W^\dagger\xi^{AB} = AM^2B^T$, $\xi^{AB}W^\dagger\xi^{AB}W^\dagger\xi^{AB} = AM^3B^T$ and so forth. We obtain the following relation:

$$\begin{aligned}
R_W(\xi) = &AB^T + AMB^T + AN_1^T B_\perp^T + A_\perp N_2 B^T - \frac{1}{8}AM^3B^T \\
&- \frac{3}{8}AM^2N_1^T B_\perp^T - \frac{3}{8}A_\perp N_2 M^2 B^T + A_\perp N_2 N_1^T B_\perp^T - A_\perp N_2 M N_1^T B_\perp^T \\
&+ \frac{1}{16}AM^3N_1^T B_\perp^T + \frac{1}{16}A_\perp N_2 M^3 B^T + \frac{1}{64}AM^4B^T + \frac{1}{4}A_\perp N_2 M^2 N_1^T B_\perp^T.
\end{aligned}$$

Collecting terms by the leftmost and rightmost factors, we obtain:

$$\begin{aligned} R_W(\xi) = & A\left(I_k + M - \frac{1}{8}M^3 + \frac{1}{64}M^4\right)B^T \\ & + A\left(I_k - \frac{3}{8}M^2 + \frac{1}{16}M^3\right)N_1^T B_\perp^T \\ & + A_\perp N_2\left(I_k - \frac{3}{8}M^2 + \frac{1}{16}M^3\right)B^T \\ & + A_\perp N_2\left(I_k - M + \frac{1}{4}M^2\right)N_1^T B_\perp^T \quad . \end{aligned}$$

Finally, treating the first and fourth lines as a polynomial expression in M, and taking its polynomial square root, we can split the above sum into the product of an $n \times k$ matrix and a $k \times m$ matrix:

$$\begin{aligned} R_W(\xi) = & \left[A\left(I_k + \frac{1}{2}M - \frac{1}{8}M^2\right) + A_\perp N_2\left(I_k - \frac{1}{2}M\right)\right] \cdot \\ & \left[B\left(I_k + \frac{1}{2}M^T - \frac{1}{8}\left(M^T\right)^2\right) + B_\perp N_1\left(I_k - \frac{1}{2}M^T\right)\right]^T . \end{aligned}$$

Appendix D. Rank One Pseudoinverse Update Rule

For completeness we develop below the procedure for updating the pseudoinverse of a rank-1 perturbed matrix (Meyer, 1973), following the derivation of Petersen and Pedersen (2008). We wish to find a matrix G such that for a given matrix A along with its pseudo-inverse $A^\dagger$, and vectors of appropriate dimension c and d, we have:

$$\left(A + cd^T\right)^\dagger = A^\dagger + G.$$

We have used the fact that A has a full column rank to simplify slightly the algorithm of Petersen and Pedersen (2008).

References

P.-A. Absil and J. Malick. Projection-like retractions on matrix manifolds. Technical Report UCL-INMA-2010.038, Department of Mathematical Engineering, Université catholique de Louvain, July 2010.

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton Univ Press, 2008.

B. Bai, J. Weston, R. Collobert, and D. Grangier. Supervised semantic indexing. *Advances in Information Retrieval*, pages 761–765, 2009.

A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6(1):937–965, 2006.

Algorithm 7 : Rank one pseudo-inverse update

Input: Matrices $A, A^\dagger \in \mathbb{R}_*^{n\times k}$, such that $A^\dagger$ is the pseudo-inverse of A, vectors $c \in \mathbb{R}^{n\times 1}$, $d \in \mathbb{R}^{k\times 1}$

Output: Matrix $Z^\dagger \in \mathbb{R}_*^{k\times n}$, such that $Z^\dagger$ is the pseudo-inverse of $A + cd^T$.

Compute:	matrix dimension
$v = A^\dagger c$	$k \times 1$
$\beta = 1 + d^T v$	1×1
$n = A^{\dagger T} d$	$n \times 1$
$\hat{n} = A^\dagger n$	$k \times 1$
$w = c - Av$	$n \times 1$
if $\beta \neq 0$ AND $\|w\| \neq 0$	
$G = \frac{1}{\beta}\hat{n}w^T$	$k \times n$
$s = \frac{\beta}{\|w\|^2\|n\|^2+\beta^2}$	1×1
$t = \frac{\|w\|^2}{\beta}\hat{n} + v$	$k \times 1$
$\hat{G} = s \cdot t \left(\frac{\|n\|^2}{\beta}w + n\right)^T$	$k \times n$
$G = G - \hat{G}$	$k \times n$
elseif $\beta = 0$ AND $\|w\| \neq 0$	
$G = -A^\dagger \frac{n}{\|n\|^2}$	$k \times 1$
$G = Gn^T$	$k \times 1$
$\hat{G} = v\frac{w^T}{\|w\|^2}$	$k \times n$
$G = G - \hat{G}$	$k \times n$
elseif $\beta \neq 0$ AND $\|w\| = 0$	
$G = -\frac{1}{\beta}vn^T$	$k \times n$
elseif $\beta = 0$ AND $\|w\| = 0$	
$\hat{v} = \frac{1}{\|v\|^2}v\left(v^T A^\dagger\right)$	$k \times n$
$\hat{n} = \frac{1}{\|n\|^2}\left(A^\dagger n\right)n^T$	$k \times n$
$G = \frac{v^T A^\dagger n}{\|v\|^2\|n\|^2}vn^T - \hat{v} - \hat{n}$	$k \times n$
endif	
$Z^\dagger = A^\dagger + G$	

J. Briët, F.M. de Oliveira Filho, and F. Vallentin. The Grothendieck problem with rank constraint. In *Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems, MTNS*, 2010.

G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11:1109–1135, 2010.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 22nd IEEE Conference on Computer Vision and Pattern*

Recognition, pages 248–255, 2009.

J. Deng, A. Berg, and L. Fei-Fei. Hierarchical Semantic Indexing for Large Scale Image Retrieval. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition*, pages 785–792, 2011.

M.P. Do Carmo. *Riemannian Geometry*. Birkhauser, 1992.

L. Eldén and B. Savas. A Newton–Grassmann method for computing the best multi-linear rank-(r1, r2, r3) approximation of a tensor. *SIAM Journal on Matrix Analysis and applications*, 31(2): 248–271, 2009.

M. Fazel. *Matrix Rank Minimization with Applications*. PhD thesis, Electrical Engineering Department, Stanford University, 2002.

M. Fazel, H. Hindi, and S. Boyd. Rank minimization and applications in system theory. In *Proceedings of the 2004 American Control Conference*, pages 3273–3278. IEEE, 2005.

A. Globerson and S. Roweis. Metric learning by collapsing classes. In *Advances in Neural Information Processing Systems*, volume 18, page 451, 2006.

J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems*, volume 17, 2005.

D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1371–1384, 2008.

M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme. *SIAM Journal on Matrix Analysis and Applications*, 32(1):115–132, 2011.

P. Jain, B. Kulis, I.S. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In *Advances in Neural Information Processing Systems*, volume 20, pages 761–768, 2008.

P. Jain, R. Meka, and I. Dhillon. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*, volume 24, pages 937–945, 2011.

M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-Rank Optimization on the Cone of Positive Semidefinite Matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010a.

M. Journée, Y. Nesterov, P. Richtárik, and R. Sepulchre. Generalized power method for sparse principal component analysis. *The Journal of Machine Learning Research*, 11:517–553, 2010b.

S.M. Kakade, S. Shalev-Shwartz, and A. Tewari. Regularization techniques for learning with matrices, 2010. Arxiv preprint arXiv:0910.0610v2.

R.H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *The Journal of Machine Learning Research*, 99:2057–2078, 2010.

B. Kulis, M.A. Sustik, and I.S. Dhillon. Low-rank kernel learning with bregman matrix divergences. *The Journal of Machine Learning Research*, 10:341–376, 2009.

K. Lang. Learning to filter netnews. In *Proceeding of the 12th Internation Conference on Machine Learning*, pages 331–339, 1995.

C.D. Manning, P. Raghavan, H. Schutze, and Ebooks Corporation. *Introduction to Information Retrieval*, volume 1. Cambridge University Press Cambridge, UK, 2008.

R. Meka, P. Jain, C. Caramanis, and I.S. Dhillon. Rank minimization via online learning. In *Proceedings of the 25th International Conference on Machine learning*, pages 656–663, 2008.

C.D. Meyer. Generalized inversion of modified matrices. *SIAM Journal on Applied Mathematics*, 24(3):315–323, 1973.

G. Meyer, S. Bonnabel, and R. Sepulchre. Regression on fixed-rank positive semidefinite matrices: a Riemannian approach. *The Journal of Machine Learning Research*, 12:593–625, 2011.

B.K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24 (2):227–234, 1995.

Sahand Negahban and Martin J. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. In *Proceedings of the 27th International Conference on Machine Learning*, pages 823–830, 2010.

C. Oberlin and S.J. Wright. Active set identification in nonlinear programming. *SIAM Journal on Optimization*, 17(2):577–605, 2007.

K.B. Petersen and M.S. Pedersen. The matrix cookbook, Oct. 2008.

B. Recht, M. Fazel, and P.A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

S. Shalev-Shwartz, Y. Singer, and A.Y. Ng. Online and batch learning of pseudo-metrics. In *Proceedings of the Twenty-first International Conference on Machine Learning*, page 94. ACM, 2004.

Uri Shalit, Daphna Weinshall, and Gal Chechik. Online learning in the manifold of low-rank matrices. In *Advances in Neural Information Processing Systems 23*, pages 2128–2136. MIT Press, 2010.

B. Vandereycken and S. Vandewalle. A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations. *SIAM Journal on Matrix Analysis and Applications*, 31(5): 2553–2579, 2010.

B. Vandereycken, P.-A. Absil, and S. Vandewalle. Embedded geometry of the set of symmetric positive semidefinite matrices of fixed rank. In *Statistical Signal Processing, 2009. SSP'09. IEEE/SP 15th Workshop on*, pages 389–392. IEEE, 2009.

K.Q. Weinberger and L.K. Saul. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009.

J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI–11)*, 2011.

E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems*, volume 15, pages 505–512. MIT Press, 2002.

L. Yang. An overview of distance metric learning. Technical report, School of Computer Science, Carnegie Mellon University, 2007.

Y. Yang and J.O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420, 1997.

2.3 Learning a functional representation of neural ISH images

BIOINFORMATICS

Vol. 29 ISMB/ECCB 2013, pages i36–i43
doi:10.1093/bioinformatics/btt207

FuncISH: learning a functional representation of neural ISH images

Noa Liscovitch[1,*,†], Uri Shalit[1,2,†] and Gal Chechik[1]

[1]Gonda Multidisciplinary Brain Research Center, Bar-Ilan University, Ramat-Gan 52900, Israel and [2]ICNC-ELSC, Hebrew University of Jerusalem, Jerusalem 91904, Israel

ABSTRACT

Motivation: High-spatial resolution imaging datasets of mammalian brains have recently become available in unprecedented amounts. Images now reveal highly complex patterns of gene expression varying on multiple scales. The challenge in analyzing these images is both in extracting the patterns that are most relevant functionally and in providing a meaningful representation that allows neuroscientists to interpret the extracted patterns.

Results: Here, we present *FuncISH*—a method to learn functional representations of neural *in situ* hybridization (ISH) images. We represent images using a histogram of local descriptors in several scales, and we use this representation to learn detectors of functional (GO) categories for every image. As a result, each image is represented as a point in a low-dimensional space whose axes correspond to meaningful functional annotations. The resulting representations define similarities between ISH images that can be easily explained by functional categories. We applied our method to the genomic set of mouse neural ISH images available at the Allen Brain Atlas, finding that most neural biological processes can be inferred from spatial expression patterns with high accuracy. Using functional representations, we predict several gene interaction properties, such as protein–protein interactions and cell-type specificity, more accurately than competing methods based on global correlations. We used FuncISH to identify similar expression patterns of GABAergic neuronal markers that were not previously identified and to infer new gene function based on image–image similarities.

Contact: noalis@gmail.com

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

In recent years, high-resolution expression data measured in mammalian brains became available in quantities and qualities never witnessed before (Henry and Hohmann, 2012; Lein *et al.*, 2007; Ng *et al.*, 2009), calling for new ways to analyze neural gene expression images. Most existing methods for bio-imaging analysis were developed to handle data with different characteristics, like *Drosophila* embryos (Frise *et al.*, 2010; Peng *et al.*, 2007; Pruteanu-Malinici *et al.*, 2011) or cellular imagery (Coelho *et al.*, 2010; Peng *et al.*, 2010). The mammalian brain, composed of billions of neurons and glia, is organized in highly complex anatomical structures and poses new challenges for analysis. Current approaches for analyzing brain images are based on smooth non-linear transformations to a reference atlas (Davis and Eddy, 2009; Hawrylycz *et al.*, 2011) and may be insensitive to fine local patterns like those emerging from the layered structure of the cerebellum or the spatial distribution of cortical interneurons.

Another challenge for automatic analysis of biological images lies in providing human interpretable analysis. Most machine-vision approaches are developed for tasks in analysis of natural images, like object recognition. In such tasks, humans can understand the scene effortlessly and infer complex relations between objects easily. In bio-imaging, however, the goal of image analysis is often to reveal features and structures that are hardly seen even by experts. It is, therefore, important that an image analysis approach provides meaningful interpretation to any patterns or structures that it detects.

Here, we develop a method to *learn functional representations of expression images* by using predefined functional ontologies. This approach has two main advantages, accuracy and interpretability, and it builds on a growing body of work in object recognition in natural images, showing how images can be represented using the activations of a large set of detectors (Deng *et al.*, 2011; Li *et al.*, 2010a, b; Malisiewicz, 2012; Malisiewicz *et al.*, 2011; Torresani *et al.*, 2010). For object recognition, the detectors may include common objects, like a detector for the presence of a chair, a mug or a door. Here, we show how to adapt this idea to represent gene expression images, by training a large set of detectors, each corresponding to a known functional category, like *axon guidance* or *glutamatergic receptors*. Once this representation is trained, every gene is represented as a point in a low-dimensional space whose axes correspond to functional *meaningful* categories.

We describe in Section 2.2 how to learn functional representations in a discriminative way and demonstrate the effectiveness of the approach on *in situ* hybridization (ISH) gene expression images of the adult mouse brain collected by the Allen Institute for Brain Science (Lein *et al.*, 2007). ISH image analysis has been used in the past to infer gene biological functions from spatial co-expression in non-neural tissues (Frise *et al.*, 2010). However, inferring functions based on gene expression patterns in the brain is believed to be hard, as several studies found very low variability between transcriptomic patterns of different brain regions, sometimes even lower than between-subject variability for the same area (Khaitovich *et al.*, 2004, 2005). Neural expression patterns are usually studied using methods that average expression values over a brain region, and this averaging removes fine-resolution spatial information that may differentiate between brain regions. Here, we analyze high-resolution ISH images at several scales, taking into account subtle, even cellular resolution, information for functional inference.

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

We find that gene function can indeed be inferred from neural ISH images, particularly in biological processes that are related to neural activities. Our approach detects related genes with better accuracy based on the similarity of their functional representations. Furthermore, these similarities can be explained and interpreted using semantic terms.

2 METHODS

2.1 The data

We used whole-brain, expression-masked images of gene expression measured using ISH, publicly available at the Allen Brain Atlas (www.brain-map.org, also see Supplementary Material). Expression was measured for the entire mouse genome. For each gene, a different adult mouse brain was sliced into 100-μm thick slices, mRNA abundance was measured and the slice was imaged. The database holds image series for >20 K transcripts. Most genes have one corresponding image series, containing ~25 imaged brain slices. Some genes were imaged more than once and have several associated image series. In our analysis, we used the most medial slice for each image series, yielding a typical image size of 8 K × 16 K pixels. In all, 4823 of the available 21 174 images showed no expression in the brain and were ignored in subsequent analysis, leaving 16 351 images representing 15 612 genes. We also tested our approach on a larger image set constructed by taking three images for each gene: the medial slice, and lateral slices at 30% and 50% of brain size (from one hemisphere). The results with this three-image set were mixed, and all results reported later in the text are for the one-slice dataset (Supplementary Material). Figure 1 shows examples of images, demonstrating the complexity of neural expression patterns across brain regions and multiple scales. The images analyzed in our study were in gray scale but are shown here as color-coded by expression intensity for better visualization.

2.2 A functional representation of images

We present a method to identify similarities between neural ISH images and to explain these similarities in functional terms.

Our method consists of a ***visual phase***, where we transform the raw pixel images into a robust visual representation, and a ***semantic phase***, where we transform that ***visual representation*** using a set of 2081 gene-function detectors. The output of these detectors comprises a higher-order ***semantic representation*** of the images in a gene-functional space (Fig. 2). Similar two-phase systems have recently been proposed and applied successfully for tasks, such as cross-domain image similarity and object detection in natural images (Deng *et al.*, 2011; Li *et al.*, 2010a, b; Malisiewicz, 2012; Malisiewicz *et al.*, 2011; Torresani *et al.*, 2010).

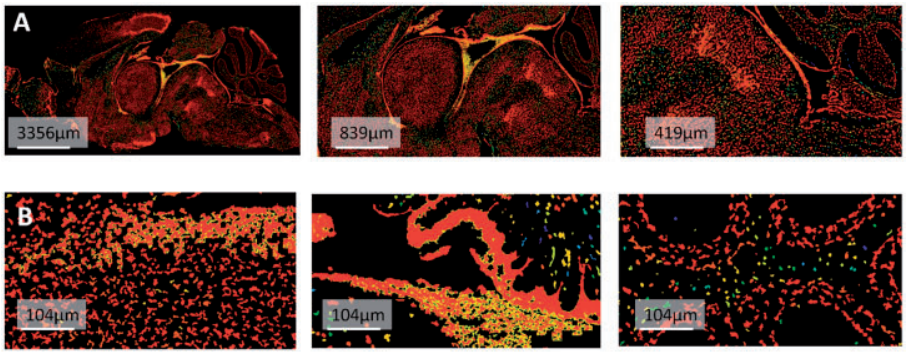


Fig. 1. The raw data. ISH image for the gene Tuba1 shown (**A**) at different scales and (**B**) in three different regions

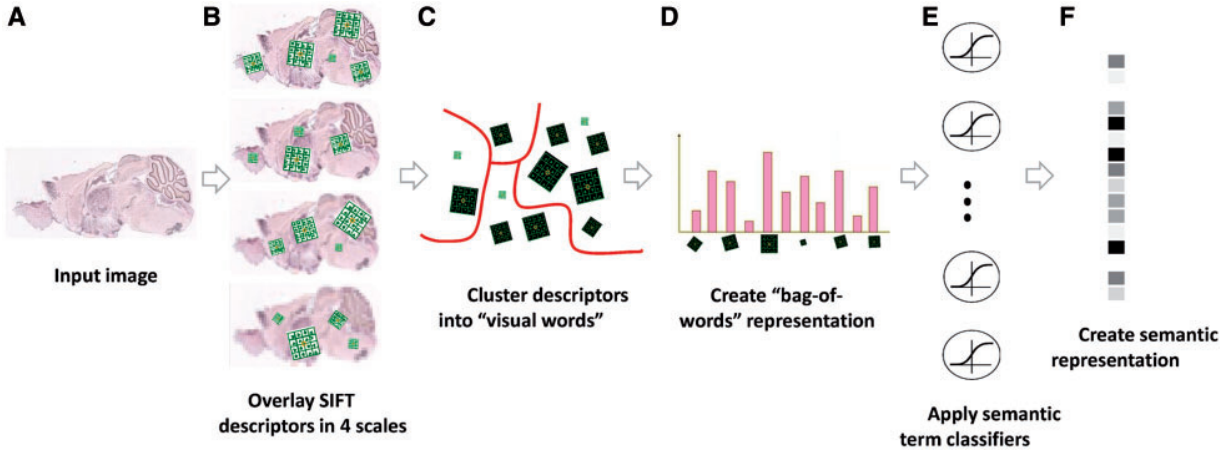


Fig. 2. Illustration of the image processing pipeline. (**A**) Original image in pixel grayscale indicating level of gene expression. (**B**) Local SIFT descriptors are extracted from image at 4 resolutions. (**C**) Descriptors from all 16351 images are clustered into 500 representative 'visual words' for each resolution level using k-Means. (**D**) Each image is represented as a histogram counting the occurrences of visual words. (**E**) L2-regularized logistic regression classifiers are applied for 2081 GO categories. (**F**) The final 2081 dimensional image representation

For the first, visual, phase, we first represent each image as a collection of local descriptors using SIFT features (Lowe, 2004). This step aims to address the problem that ISH brain images of the same gene vary significantly in shape and size when measured in different brains (Kirsch *et al.*, 2012). SIFT features are histograms of oriented gradients on a small grid. The resulting image-patch SIFT descriptor is invariant to small rotation and illumination (but not to scale), making imaged-slices from different brains more comparable. We computed SIFT descriptors of dimension 128 extracted on a dense grid spanning the full image (Bosch *et al.*, 2006, 2007; Csurka and Dance, 2004), at four spatial resolutions. In ISH images, different information lies in different descriptor sizes, and we wish that the representation captures spatial patterns both at the level of single cells, micro-circuitry and at the coarser level of distribution of expression across brain layers. To capture information at multiple scales, we used the VLFeat implementation of SIFT (Vedaldi and Fulkerson, 2010), where scale-invariance is not incorporated automatically. Specifically, each image is represented as a collection of ~1 M SIFT descriptors, computed by down sampling each image at a factor of 1, 2, 4 and 8. As the descriptors were extracted from high-resolution images, which are mostly dark, many descriptors were completely dark and were discarded.

Next, to achieve a compact non-linear representation of each image, we aggregate the descriptors from all images for a given resolution level and cluster them to form a dictionary of distinct 'visual words' per each resolution level. We used the original Lloyd optimization for k-Means with L_2 distance, initializing the centroids by randomly sampling data points. The clustering procedure was repeated multiple times ($n = 3$), and the solution with the lowest energy was used. We tested four different dictionary sizes ($k = 100$, 200, 500 and 1000), all yielding similar results (Supplementary Material), and we report later in the text results for $k = 500$, which obtained slightly higher accuracies. Next, we construct a standard 'bag-of-words'[20,21] description of each image. As a result of this process, each image is described by four concatenated 500-dimensional vectors counting how many times each 'visual word' appeared in it at a given resolution level. We also added a count of the number of zero descriptors per resolution level, ending up with a 2004-dimensional vector describing each image. Using this approach, similar spatial information from different brain regions is preserved, as opposed to using global correlation-based approaches.

We then turn to the second, 'semantic', phase, and represent each image by a set of functional descriptors. Given a set of predefined Gene Ontology (GO) annotations of each gene, we train one separate classifier for each known biological annotation category, using the SIFT bag-of-words representation as an input vector. Specifically, here, we trained a set of 2081 L_2-regularized logistic regression classifiers [using LIBLINEAR (Fan *et al.*, 2008)] corresponding to biological-processes GO classes that have 15–500 annotated genes (Supplementary Material). We trained the classifiers using two layers of 5-fold cross-validation, performed as follows: the full set of 16 351 gene images was split into five non-overlapping equal sets (without controlling for the number of positives in each split), training the classifiers on four of them and testing performance on the fifth unseen test set of images. This procedure was repeated five times, each time with a different set acting as the test set. All accuracy and other results later in the text are reported for a held-out test set that was not used during training.

To tune the logistic regression regularization hyperparameter, we used a second layer of cross-validation. We repeated the splitting procedure within each of the five training sets, splitting each of them again into five subsets of images, using four for training and the fifth as a validation set. The regularization hyperparameter was selected from the values (0.001, 0.01, 0.1, 1, 10 and 100). At the end of this process, each gene is then represented as a vector of 'activations', corresponding to the likelihood that the gene belongs to one functional category, such as '*forebrain development*' or '*regulation of fatty acid transport*'.

The representation described earlier in the text removes important information about global location in the brain. We, therefore, also tested an approach using spatial pyramids (Lazebnik *et al.*, 2006), where descriptor histograms are computed separately for different parts of the image. Unfortunately, this approach results in feature vectors whose dimensionality was too high for the current dataset and yielded poor classification results (Supplementary Material).

2.3 Similarity between functional profiles

We use two gene–gene similarity measures in this work, taking each gene as a vector of functional category activations. The first, *flat-sim*, is simply the linear correlation of two functional category activation vectors. The second, *GO-sim*, takes into account the known directed acyclic graph (DAG) structure among the functional categories of the GO annotation.

Formally, the *flat-sim* score between a pair of L_2-normalized feature vectors $a = (a_1 \dots a_m)$ and $b = (b_1 \dots b_m)$ is given by their dot product $\textit{flat-sim}\ (a, b) = \sum_{i=1}^{m} a_i \cdot b_i$. This additive similarity measure allows assessing the contribution of each individual feature to the overall similarity score, by setting the contribution of the feature i (corresponding to GO category i) to $a_i \cdot b_i$. Thus, for each pair of similar images, we can sort the GO categories by order of their contribution to the similarity, providing a semantic interpretation of the correlation.

However, *flat-sim* does not take into account that the activation of some functional categories can be far more informative than others. For example, two genes that share a specific function like '*negative regulation of systemic arterial blood pressure*' are much more likely to be functionally similar than a pair of genes sharing a more general category like '*metabolism*'. We address this issue by adapting a functional similarity measure between gene products developed by (Schlicker *et al.*, 2006), which we refer to as *GO-sim*. GO-sim is designed to give high similarity scores to gene pairs that share many specific and similar functional categories. We treat our model's functional activations as binary annotations (using a threshold of 0.5) and calculate *GO-sim* as follows.

For each GO category i, we calculate its *information content* (*IC*) as $IC(i) = -log_{10} \frac{\#genes\ in\ i}{total\ \#\ of\ genes}$, which measures the specificity of each category. For each pair of categories i and j, we consider the set of their common ancestors $anc(i,j)$ and define $sim_{rel}(i,j) = \max_{k \in anc(i,j)} \frac{2IC(k)}{IC(i)+IC(j)}(1 - 10^{-IC(k)})$. The measure sim_{rel} is symmetric, bounded between 0 and 1, and attains larger values for pairs of categories that are both specific *and* close to each other in the GO graph.

In our method, each gene is annotated with multiple categories. Naïvely, we could calculate the mean sim_{rel} measure between all pairs of categories, but calculating this mean could give weight to many irrelevant categories and be sensitive to the addition of extra annotations to a gene. Instead, we use a more robust method to measure similarity between two sets of function annotations, developed by (Schlicker *et al.*, 2006). This method relies on the most similar gene pairs, instead of all the pairs. For two **binary** activation vectors $a = (a_1 \dots a_m)$, $b = (b_1 \dots b_m)$ define a matrix $S_{ij} = sim_{rel}(i, j)a_i b_j$. Then we define $sim_{a \to b} = \frac{1}{m}\sum_{i=1}^{m}(\max_{j=1 \dots m} S_{ij})$ that measures for each annotation of a its most similar annotation in b and averages across all of a 's annotations. We similarly define $sim_{b \to a}$ with the roles of a and b switched, and use it to define $GO\text{-}sim = \max(sim_{a \to b}, sim_{b \to a})$. To assess the contribution of individual gene functional annotations to the *GO-sim* measure, we look at the category pairs (i,j) corresponding to the highest values of S_{ij}. Each such pair also has its 'most informative common ancestor' $MICA(i,j) = \operatorname{argmax}_{k \in anc(i,j)} \frac{2IC(k)}{IC(i)+IC(j)}(1 - 10^{-IC(k)})$. These ancestor functional categories give a succinct interpretation of the similarity between genes a and b.

Computing *GO-sim* for $n = 16\,351$ genes, each with m functional annotations, is computationally burdensome, requiring $O(n^2m^2)$

operations. In this study, we, therefore, use only 164 brain-related categories of the 2081 functional categories for calculating *GO-sim*.

3 RESULTS

We start with evaluating the quality of the low-dimensional semantic representation that we learned in two aspects: the classification accuracy for individual semantic terms and the precision of our gene–gene similarity measure compared with a spatial correlation-based method. We then take a closer look at discriminative spatial patterns, mapping them back onto raw images. Finally, we use the geometry of the low-dimensional semantic space to infer new gene functions via gene similarities and their interpretations.

3.1 Predicting functional annotations using brain ISH images

We applied FuncISH to 16 K ISH images of 15 K genes, and we mapped each image to a vector corresponding to 2000 GO categories as functional features. We used the area under the ROC curve (AUC) as a measure of classification accuracy. All evaluations were performed on a separate held-out test set. We find that 37% of the GO categories tested yielded a test set AUC value that was significantly above random (permutation test, $P<0.05$). This was encouraging, as the variability of expression between brain regions was previously shown to be very low (Khaitovich *et al.*, 2004, 2005). This suggests that fine spatial resolution in neural tissues can reveal highly meaningful expression patterns.

Which functional categories can be best predicted by ISH images? Table 1 lists the top 15 GO categories that achieved the best test-set AUC classification scores. Interestingly, these include mostly biosynthesis/metabolism processes and neural processes. To further test whether neural categories achieve higher classification values based on neural expression patterns, Figure 3 compares the AUC scores of 164 categories related to the nervous system with the AUC scores of the remaining categories. As expected, neural GO categories receive significantly higher AUCs (Wilcoxon, $P<10^{-38}$), with 69% of categories yielding significantly above random AUC values.

These AUC values suggest that when a gene is represented as a feature vector of classifiers activations, many of the features carry a meaningful signal. The axes of the new low-dimensional representation correspond to functional properties of each gene, linking functions of the genes to the geometry of the space in which they are embedded.

3.2 Comparison with Neuroblast, the ABA image-correlation tool

How well does FuncISH compare with other methods suggested for finding similarity between these images? We compared our results with *NeuroBlast,* a method to detect image–image similarities available on the ABA website (Hawrylycz *et al.*, 2011). This method uses a non-linear mapping of the images to a reference anatomical atlas to apply voxel–voxel correlation between the images.

To evaluate the quality of the similarity measure, we used three sets of pairwise relations as evidence of gene relatedness: (i) markers of known cell types (Cahoy *et al.*, 2008), such as astrocytes or oligodendrocytes; (ii) occurrence in the same KEGG pathway (Kanehisa, 2002); and (iii) a set of known protein–protein interactions taken from *IntAct* (Kerrien *et al.*, 2012). For each of the 16 531 genes, we ranked the 100 most similar genes according to four different similarity measures: (i) FuncISH GO-sim, (ii) FuncISH flat-sim, (iii) cosine similarity between the SIFT bag-of-words representations (Fig. 2D) and (iv) the ABA *NeuroBlast* tool. For each of the pairwise relations (cell-type markers, KEGG pathway and PPIs), we plot the mean fraction of relations retrieved at the top-K most similar genes (precision-at-k), a standard method in information retrieval (Manning and Raghavan, 2009). Figure 4 shows that for all three validation labels, FuncISH *GO-sim* provides superior precision for the top 10 ranked similar genes. The superior precision of GO-sim over flat-sim is presumably because

Table 1. The GO categories classified with highest test-set AUC values

GO ID	GO category name	No. of genes	AUC
GO:0060311	Negative regulation of elastin catabolic process	17	1
GO:0042759	Long-chain fatty acid biosynthetic process	23	0.98
GO:0009449	γ-Aminobutyric acid biosynthetic process	20	0.96
GO:0009448	γ-Aminobutyric acid metabolic process	23	0.96
GO:0032348	Negative reg. of aldosterone biosynthetic process	21	0.94
GO:2000065	Negative regulation of cortisol biosynthetic process	21	0.94
GO:0043206	Fibril organization	23	0.94
GO:0031947	Negative reg. of glucocorticoid biosynthetic process	22	0.94
GO:0042136	Neurotransmitter biosynthetic process	23	0.94
GO:0022010	Central nervous system myelination	29	0.89
GO:0008038	Neuron recognition	20	0.87
GO:0042220	Response to cocaine	30	0.87
GO:0050919	Negative chemotaxis	16	0.86
GO:0042274	Ribosomal small subunit biogenesis	15	0.86
GO:0016486	Peptide hormone processing	17	0.85

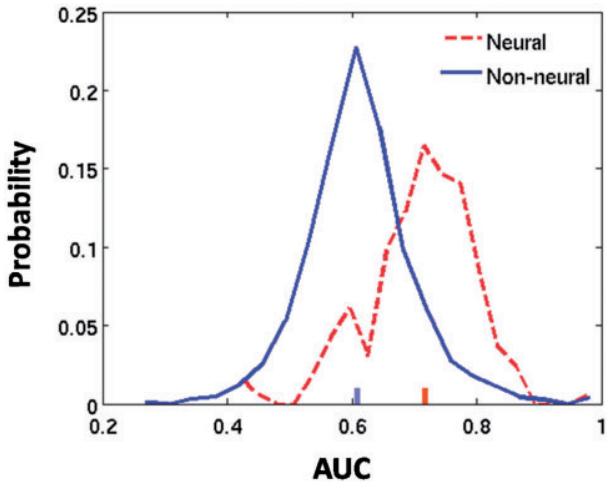


Fig. 3. AUC scores for GO categories related to the nervous system (dashed, red) and the remaining categories (solid, blue). AUC scores are significantly higher for neural categories (Wilcoxon test, $p<10^{-38}$). The red and blue ticks indicate the median of each set

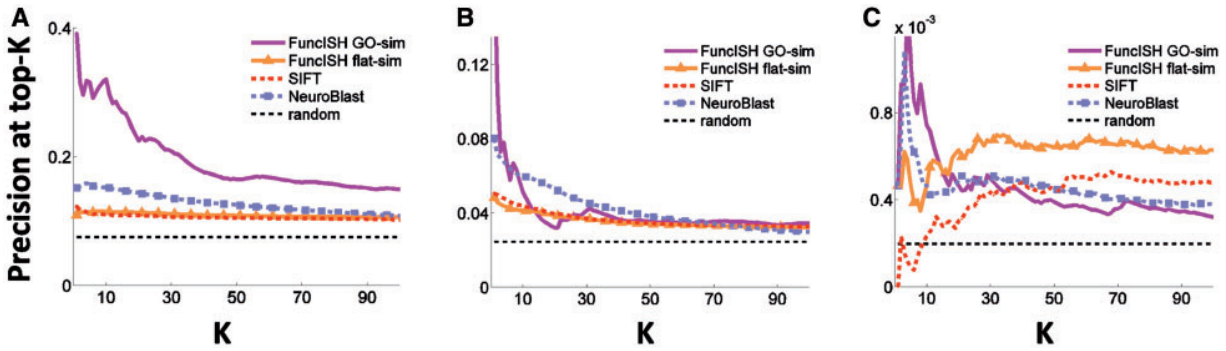


Fig. 4. Precision at top-K for similarity defined by (**A**) cell type marker (**B**) KEGG pathways (**C**) protein–protein interaction. Precision was measured using functional representations (*FuncISH*, purple lines for *GO-sim*, orange for *flat-sim*), SIFT (red) and NeuroBlast (blue)

GO-sim weighs categories more correctly and also possibly because GO-sim was limited to brain-related categories that tend to be more accurately predicted (Fig. 3). On the other hand, we see that NeuroBlast outperforms flat-sim in most cases.

3.3 Identifying and explaining similarities between GABAergic neuron markers

We now turn to a deeper look into the similarity predictions. Interestingly, the highest classification scores were achieved for the neural-related categories *GABA biosynthetic process* and *GABA metabolic process* (shown in Table 1), implying that our algorithm can identify spatial patterns of GABAergic neurons. A prominent member of the GABAergic neuron marker family is *parvalbumin B* (*Pvalb*), which encodes for a calcium-binding protein. We examined the genes that are most similar to *Pvalb*, and we found that another GABAergic neuronal marker and a calcium-binding protein, *calbindin D28K* (*Calb1*), is at the top 15 most similar gene lists for all associated image series. *Pvalb* and *Calb1* belong to a family of cellular Ca^{2+} buffers in GABAergic interneurons. The third member in this family is calretinin (*Calb2*). Looking at the similarity rank of *Calb1* and *Calb2*, *Calb2* ranks at the top 2 percentile (of 16 351 images in the dataset) at 16 of 17 cases. Similarities between these three genes were not identified by NeuroBlast. This may be because NeuroBlast uses spatial correlation measures that produce results heavily reliant on the spatial location of expression, whereas FuncISH can identify patterns that can appear in different regions of the brain. A major benefit of representing genes in the functional embedding space is that similarities between genes can be 'explained' in functional terms. *Calb1*, *Pvalb* and *Calb2* are all involved in regulation of synaptic plasticity (Schwaller, 2012). When looking at the semantic interpretations explaining the similarities between the genes, 6 of the top 10 GO categories are indeed directly related to synaptic plasticity, such as '*synaptic transmission*', '*regulation of synaptic plasticity*' and '*learning*'.

3.4 Finding important spatial patterns in different scales using SIFT 'visual words'

A major advantage of representing ISH images with SIFT descriptors is the ability to point directly to spatial patterns in these complex images. Although their name suggest differently, SIFT descriptors at several scales capture different types of patterns. Figure 5 shows three visual words for each of the four scales, selected as the visual words that contributed most to classification. Scale invariance is often assumed when analyzing natural images, as objects are photographed at varying distances. ISH images, however, contain distinctive information in the different scales. As Figure 5 demonstrates, the four sizes of visual words correspond to grids capturing different neural entities. The smallest descriptors cover an actual area of $36 \times 36\,\mu m^2$ and capture fine-scaled information, such as cell shapes and cell densities; the medium-size discriminative descriptors of $72 \times 72\,\mu m^2$ tend to trace thinner cell layers; larger descriptor sizes of $144 \times 144\,\mu m^2$ and $288 \times 288\,\mu m^2$ can cover large and intricate patterns of a mixture of cells and cell types in a tissue. Interestingly, the four visual words with the highest contribution to classification were the words counting the zero descriptors in each scale. This means that the highest information content lies in 'least informative' descriptors, and that overall expression levels ('sparseness' of expression) are important factors in functional prediction of genes based on their spatial expression. Our method presents a new representation of ISH imagery as SIFT descriptors, and using multiple scales allows revealing the multiresolution nature of the images.

Which scale carries the most meaningful signal for functional prediction? Figure 5E shows the mean absolute value of visual words weights in every scale for all GO categories, showing that all scales contribute significantly to the scores, with the medium contributing most.

Figure 5A–D shows descriptors that contributed to classification of **all** the categories. Furthermore, each GO category has its own visual words that are important to its classification, and looking into their details reveals spatial properties that are unique to specific biological processes.

As an interesting example of this effect, we considered the gene *adducin β* (*Add2*). *Add2* is annotated to several GO categories, including '*positive regulation of protein binding*' and '*actin filament bundle assembly*'. Figure 6 overlays the top weighted visual words of the two categories over the *Add2* ISH image. It is easy to see that the descriptors important for classification of '*actin filament bundle assembly*' are much smaller than those important for classification of the more general

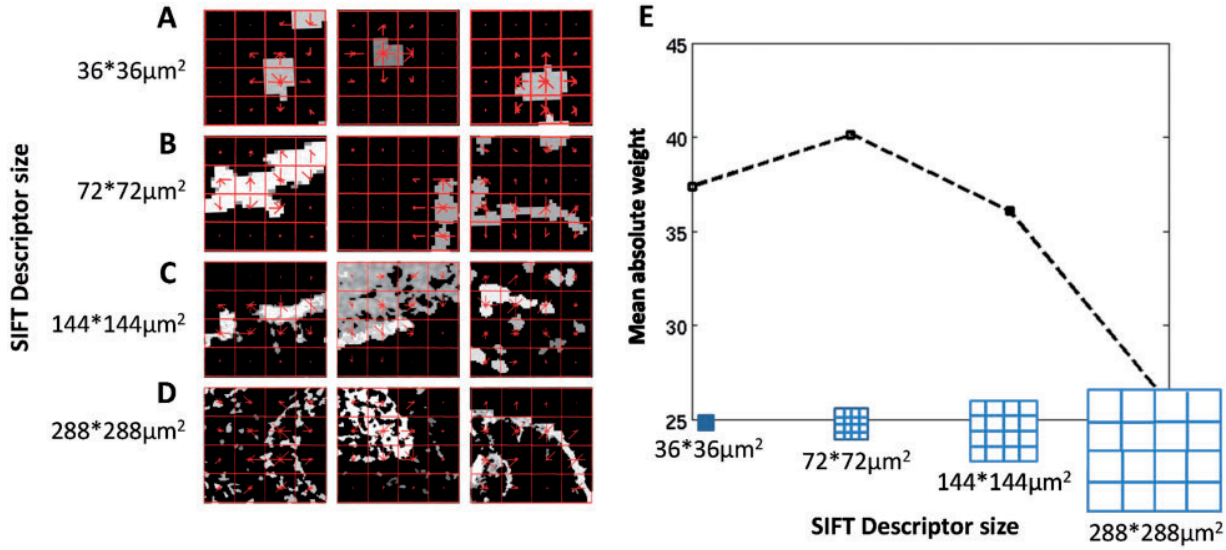


Fig. 5. Representing ISH images with visual words. (**A**, **B**, **C**, **D**) The three visual words with highest absolute weight (averaged over all categories) at each scale. The SIFT descriptors (red grid) are plotted on top of each panel. The histogram of oriented gradients used in the SIFT descriptor is plotted in the center of each element of the grid,as a set of red lines, where the length of the line correspond to the magnitude of the gradient in its direction. (**E**) Mean absolute weight for the four scales of visual words calculated over classifiers for all categories

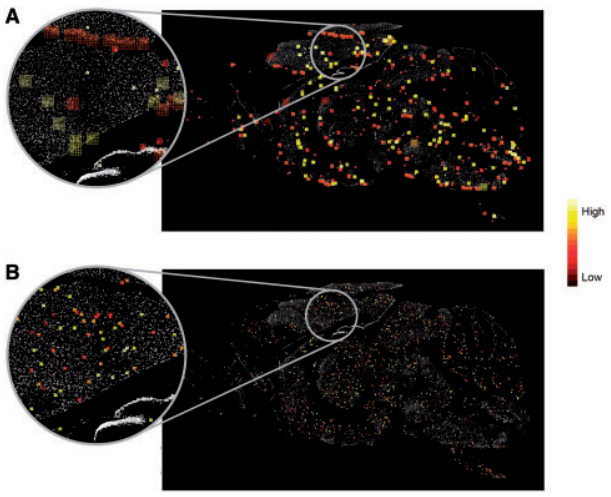


Fig. 6. The visual words important in classifying Add2 GO categories are overlaid on the Add2 ISH image. Larger descriptors are needed for the classification of 'regulation of protein binding' (**A**), while the discriminative visual words for 'actin filament bundle assembly' (**B**) are much smaller, capturing properties such as cell shapes. The descriptors are color-coded by their importance in classification, highest importance is in bright yellow

category '*positive regulation of protein binding*' (*t*-test, $P<10^{-17}$). This implies that small-scaled features, such as specific cell shapes, are important to identify genes related to actin filament bundle assembly processes. Actin assemblies are important for the navigation of neural growth cones, by re-orienting growth cones away from inhibitory cues (Challacombe *et al*., 1996). Representing the images with histograms of oriented gradients could capture tiny differences in cell shapes that are in the process of synapse formation, a developmental process occurring continuously throughout adulthood (Vidal-Sanz *et al*., 1987).

3.4 Inferring new gene functions via explainable similarities

We now demonstrate how the semantic representation learned by FuncISH can be used to propose new gene functional annotations. Consider as an example the gene *synaptopodin 2* (*Synpo2*) that is known to bind actin, but otherwise has little known associated information. FuncISH can be used to propose functional annotations for *synpo2* by looking at the genes that are similar to *Synpo2* and considering both the GO functions that contribute to this similarity and the spatial pattern of expression.

First, we find that *Synpo2* is similar to two other genes *Npepps* and *Rasa4*, but for different reasons (the list of top five semantic explanations for these similarities is shown in Table 2). *Npepps* is an aminopeptidase that is active specifically in the brain (Hui, 2007), and the similarity between *Synpo2* and *Npepps* is explained by processes related to protein processing, such as ubiquitination and protein proteolysis. At the same time, *Rasa4* is a GTPase-activating protein that suppresses the Ras/mitogen-activated protein kinase pathway in response to Ca^{2+} (Vigil *et al*., 2010), and the similarity between *Synpo2* and *Rasa4* is explained by high-level neural processes, such as axon guidance or synaptic transmission.

Interestingly, *Synpo2* and *Rasa4* are expressed in different brain regions: looking at their spatial expression patterns reveals that *Synpo2* is expressed exclusively in the thalamus, whereas *Rasa4* is expressed in olfactory areas. Therefore, their similarity is not in their global expression patterns across regions, but rather in local spatial patterns. This could reflect expression in

Table 2. Top 10 GO annotations explaining the similarities between the gene *Synpo2* and *Npepps* (left column) and *Rasa4* (right column)

Synpo2–Npepps		*Synpo2–Rasa4*	
GO ID	GO name	GO ID	GO name
GO:0070646	Protein modification by small protein removal	GO:0006836	Neurotransmitter transport
GO:0006412	Translation	GO:0051970	Negative regulation of transmission of nerve impulse
GO:0016567	Protein ubiquitination	GO:0050805	Negative regulation of synaptic transmission
GO:0051603	Proteolysis involved in cellular protein catabolic process	GO:0007411	Axon guidance
GO:0032446	Protein modification by small protein conjugation	GO:0031645	Negative regulation of neurological system process

similar cell types or tissues that exhibit similar spatial distribution at different brain regions. *Npepps* is more ubiquitously expressed in the brain, and it is located in the thalamic area where *synpo2* is expressed. The co-location of *Synpo2* and *Npepps* suggests they could be participating in similar biological processes in these areas, possibly in protein-modification processes as suggested by the list of top explanations for the similarity.

4 SUMMARY

We present *FuncISH*—a method to learn functional representations of neural ISH images, yielding an interpretable measure of similarity between complex images that are difficult to analyze and interpret. Using FuncISH, we successfully infer ~700 functional annotations from neural ISH images, and we use them to detect gene–gene similarities. This approach reveals similarities that are not captured by previous global correlation-based methods, but it also ignores important global location information. Combining local and global patterns of expression is, therefore, an important topic for further research, as well as the use of more sophisticated non-linear classifiers, such as kernel-SVM, for creating better representations. Importantly, FuncISH provides semantic interpretations for similarity, enabling the inference of new gene functions from spatial co-expression.

ACKNOWLEDGEMENTS

The authors are grateful to the Allen Institute for Brain Science for making their data available to the scientific community and for helping us with any questions. They are also grateful to Lior Kirsch for valuable discussions and technical help. They thank the reviewers for their helpful comments and suggestions.

Funding: Israeli Science Foundation (1008/09 and 1090/12 to G.C.); Marie Curie Reintegration Grant (PIRG06-GA-2009-256566 to G.C.); Google Europe Fellowship in Machine Learning (to U.S.).

Conflict of Interest: none declared.

REFERENCES

Bosch,A. *et al.* (2006) Scene classification via pLSA. *Computer Vision-ECCV 2006*, **3954**, 517–530.

Bosch,A. *et al.* (2007) Image classification using random forests and ferns. *IEEE 11th Int. Conf. Comput. Vis.*, **23**, 1–8.

Cahoy,J.D. *et al.* (2008) A transcriptome database for astrocytes, neurons, and oligodendrocytes: a new resource for understanding brain development and function. *J. Neurosci.*, **28**, 264–278.

Challacombe,J.F. *et al.* (1996) Actin filament bundles are required for microtubule reorientation during growth cone turning to avoid an inhibitory guidance cue. *J. Cell Sci.*, **109** (Pt 8), 2031–2040.

Coelho,L.P. *et al.* (2010) Quantifying the distribution of probes between subcellular locations using unsupervised pattern unmixing. *Bioinformatics*, **26**, i7–i12.

Csurka,G. and Dance,C. (2004) Visual categorization with bags of keypoints. In: *Workshop on Statistical Learning in Computer Vision, ECCV*. Vol. 1, p. 22.

Davis,F.P. and Eddy,S.R. (2009) A tool for identification of genes expressed in patterns of interest using the Allen Brain Atlas. *Bioinformatics*, **25**, 1647–1654.

Deng,J. *et al.* (2011) Hierarchical semantic indexing for large scale image retrieval. *CVPR 2011*, 785–792.

Fan *et al.* (2008) LIBLINEAR: a library for large linear classification. *J. Mach. Learn. Res.*, **9**, 1871–1874.

Frise,E. *et al.* (2010) Systematic image-driven analysis of the spatial *Drosophila* embryonic expression landscape. *Mol. Syst. Biol.*, **6**, 345.

Hawrylycz,M. *et al.* (2011) Multi-scale correlation structure of gene expression in the brain. *Neural Netw.*, **24**, 933–942.

Henry,A.M. and Hohmann,J.G. (2012) High-resolution gene expression atlases for adult and developing mouse brain and spinal cord. *Mamm. Genome*, **23**, 539–549.

Hui,K.-S. (2007) Brain-specific aminopeptidase: from enkephalinase to protector against neurodegeneration. *Neurochem. Res.*, **32**, 2062–2071.

Kanehisa,M. (2002) The KEGG database. *Novartis Found. Symp.*, **247**, 91–101; discussion 101–103, 119–128, 244–252.

Kerrien,S. *et al.* (2012) The IntAct molecular interaction database in 2012. *Nucleic Acids Res.*, **40**, D841–D846.

Khaitovich,P. *et al.* (2004) Regional patterns of gene expression in human and chimpanzee brains. *Genome Res.*, **14**, 1462–1473.

Khaitovich,P. *et al.* (2005) Parallel patterns of evolution in the genomes and transcriptomes of humans and chimpanzees. *Science*, **309**, 1850–1854.

Kirsch,L. *et al.* (2012) Localizing genes to cerebellar layers by classifying ISH images. *PLoS Comput. Biol.*, **8**, e1002790.

Lazebnik,S. *et al.* (2006) Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. *IEEE Conf. Comput. Vis. and Pattern Recognition*. Vol. 2 CVPR06, **2**, 2169–2178.

Lein,E.S. *et al.* (2007) Genome-wide atlas of gene expression in the adult mouse brain. *Nature*, **445**, 168–176.

Li,L. *et al.* (2010a) Object bank: a high-level image representation for scene classification and semantic feature sparsification. *Proc. Neural Inf. Process. Syst. 2010*, 1–9.

Li,L. *et al.* (2012) Objects as attributes for scene classification. In: *Trends and Topics in Computer Vision*. Springer, Berlin Heidelberg, pp. 57–69.

Lowe,D.G. (2004) Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, **60**, 91–110.

Malisiewicz,T. (2012) Exemplar-based representations for object detection, association and beyond, PhD Thesis.

Malisiewicz,T. *et al.* (2011) Ensemble of exemplar-SVMs for object detection and beyond. In: *2011 International Conference on Computer Vision*, 89–96.

Manning,C.D. and Raghavan,P. (2008) *Introduction to Information Retrieval*. Vol. 1, Cambridge University Press, Cambridge.

Ng,L. *et al.* (2009) An anatomic gene expression atlas of the adult mouse brain. *Nat. Neurosci.*, **12**, 356–362.
Peng,H. *et al.* (2007) Automatic image analysis for gene expression patterns of fly embryos. *BMC Cell Biol.*, **8**, S7.
Peng,T. *et al.* (2010) Determining the distribution of probes between different subcellular locations through automated unmixing of subcellular patterns. *Proc. Natl. Acad. Sci. USA*, **107**, 2944–2949.
Pruteanu-Malinici,I. *et al.* (2011) Automatic annotation of spatial expression patterns via sparse Bayesian factor models. *PLoS Comput. Biol.*, **7**, e1002098.
Schlicker,A. *et al.* (2006) A new measure for functional similarity of gene products based on Gene Ontology. *BMC Bioinformatics*, **7**, 302.
Schwaller,B. (2012) The use of transgenic mouse models to reveal the functions of Ca2+ buffer proteins in excitable cells. *Biochim. Biophys. Acta*, **1820**, 1294–1303.
Torresani,L. *et al.* (2010) Efficient object category recognition using classemes. *Comput. Vis.–ECCV 2010*, 776–789.
Vedaldi,A. and Fulkerson,B. (2010) VLFeat—an open and portable library of computer vision algorithms. *Design*, **3**, 1–4.
Vidal-Sanz,M. *et al.* (1987) Axonal regeneration and synapse formation in the superior colliculus by retinal ganglion cells in the adult rat. *J. Neurosci.*, **7**, 2894–2909.
Vigil,D. *et al.* (2010) Ras superfamily GEFs and GAPs: validated and tractable targets for cancer therapy? *Nat. Rev. Cancer*, **10**, 842–857.

Supplementary material

Using expression-masked images

Images in the Allen dataset are provided in two formats: the raw imagery, and images that were processed as previously described[1] to remove the background, yielding expression-masked images. The analysis was applied to the masked images. This is a big advantage when examining expression patterns, as noise effects coming from cytoarchitecture and underlying brain structures is reduced. Examples of a pair of images are given below in Fig S1.

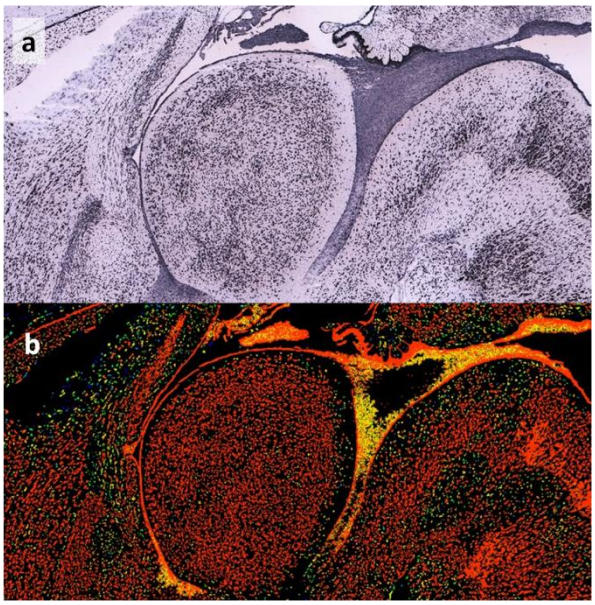


Figure S1: Regular (a) and expression-masked (b) examples of ISH images as provided by the Allen Brain Atlas, for the gene Tuba1. While the expression masked images are presented in color, the color images are in fact derived from gray-scale images, which we have used in this work.

Robustness of bag-of-words representations

In order to validate the stability of the bag-of-words gene representations, we measured the similarities between pairs of representations of images that are of the same gene but from different image series, and the similarities between the representations of different genes.

Similarity is much higher for representations of the same gene (Wilcoxon difference of medians test, $p<10^{-200}$). The similarity values are shown in figure S2. This implies that representations of the same gene, derived from different image series are indeed stable and are representative of the gene.

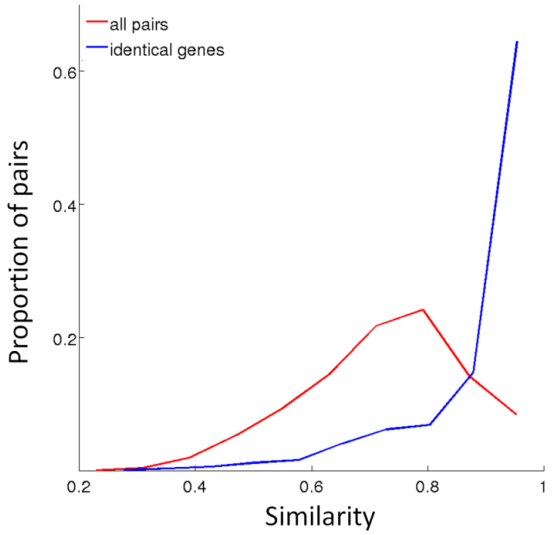


Figure S2: The similarity in the representation of same-gene pairs (blue) and different-gene pairs (red). Each curve shows the histogram of similarity values. Same-gene image series have highly similar representations.

Choosing the dictionary size

In order to choose the size of the visual word dictionary, we performed analysis with four dictionary sizes: 100, 200, 500 and 1000. Figure S3 shows mean test-set AUC values obtained using the different dictionary sizes. Mean AUC across categories is insensitive to the size of the dictionary (K). To check how stable the representations are between the different K's, we measured the Pearson correlation between AUC values of the 2081 GO categories using the different dictionary sizes. Correlation values are very high and are shown in table R1. The lowest correlation value is 0.846, between K=100 and K=1000, and is still highly significant ($P<10^{-100}$). Correspondence between AUC values for the 2081 GO categories obtained using the two dictionary sizes are shown in figure S4, showing indeed a high linear correspondence.

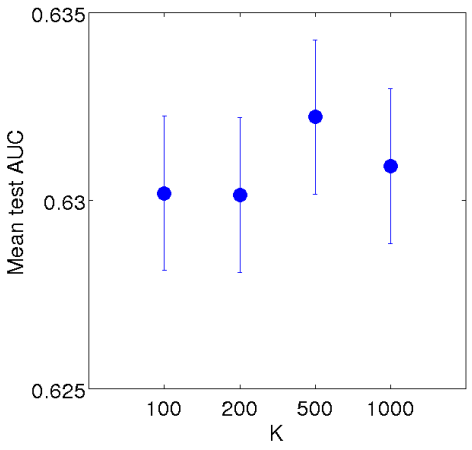


Figure S3: Mean test-AUC values for dictionary size K=100, 200, 500, 1000. Error bars indicate standard error of mean across five folds in cross-validation data.

Dictionary size (K)	100	200	500	1000
100	1	0.896	0.861	0.846
200	0.896	1	0.896	0.883
500	0.861	0.896	1	0.917
1000	0.846	0.883	0.917	1

Table S1: Pearson's rho correlation values between AUC results for 2081 categories, compared across the 4 different dictionary sizes. Correlations are high (the lowest is 0.846 between K=100 and K=1000)

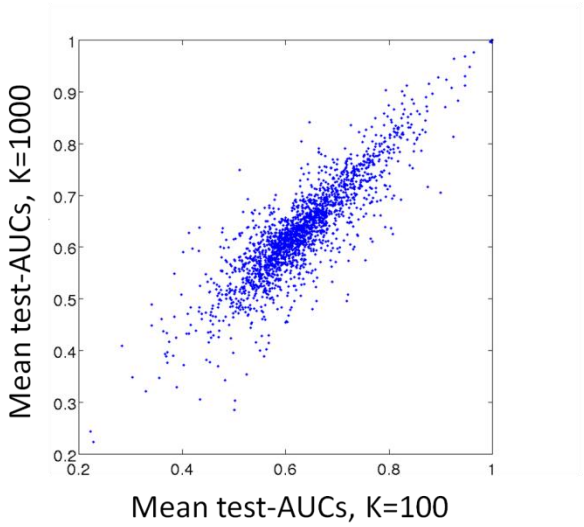


Figure S4*:* Mean test-set AUCs for dictionary size K=100 versus K=1000. This pair of dictionary sizes is the least correlated among all dictionary size pairs. It can be seen that even in this case, the correlation is high and indicative of a stable representation.

Choice of GO category size:

We chose GO categories with a number of annotations ranging from 15 to 500 genes. We set the lower limit to 15 in order to provide enough positive examples for testing the classifiers across five cross-validation partitions. The higher limit is set to 500 to preclude the resulting semantic explanations from being very general (we use more specific categories such as "*regulation of long-term neuronal synaptic plasticity*" or "*glutamate receptor signaling pathway"* and avoid general categories such as *"transport"* or *"biological regulation"*).

To make sure that this choice of categories did not cause a bias in the classification results, we checked the relation between category size and test-set AUC scores. No significant relation between the size of the GO category and the resulting AUC values (Figure S5).

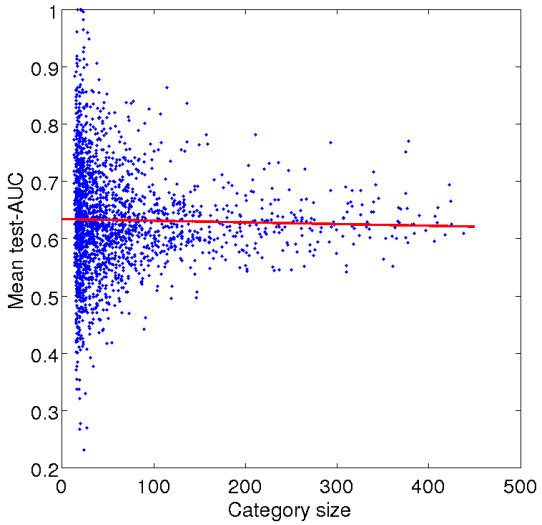


Figure S5: Mean AUC (averaged over test-splits) for the GO categories vs. GO category size (number of genes in the category). There's no significant relation between classification success of a category and the number of genes annotated to it.

Using several slices from each image series

In order to take into fuller account the 3D structure of the brain, we repeated the full set of our experiments while including two additional sagittal sections. The three sections used were taken from one hemisphere, capturing the medial section and also the 30% and 50% marks on the medial-lateral axis. An example of three such slices is shown in Figure S6.

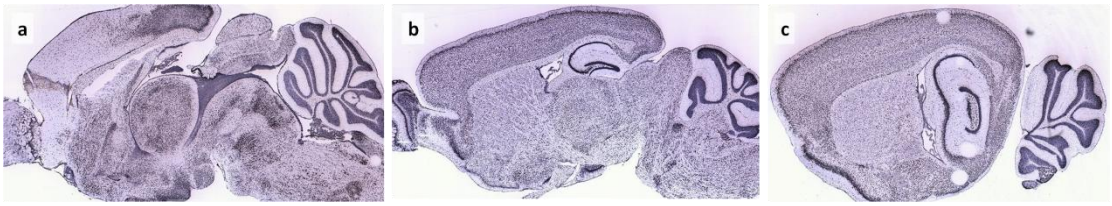


Figure S6: Each image series was represented with three slices, the most medial (a), and the 30% (b) and 50% (c) marks on the medial-lateral axis.

The results of the experiments using multiple slices were inconclusive. In some measures of performance, such as the correlation of our funcISH scores with known PPI interactions, adding more slices has improved the correlations. In others, such as correlations with cell types and pathways, the performance measures did not improve and even deteriorated slightly. The reasons for this inconsistency could be that the location of the non-medial slices is more variable, due to variation across

brains. We note that in the main paper we report the results using a single medial slice.

Applying a spatial pyramid kernel to the images

A major goal of brain-image analysis is to develop a representation that captures both low level texture and gross-anatomy structure. While the visual bag-of-words representation we have used in our work removes global structures, a main advantage is the ability to find small-scaled spatial patterns that are location-independent in the brain.

To combine local patterns with global structures in the same representation, we tested a representation of the data using spatial pyramid kernels[2]. In this approach, every image is split into 4 and 16 rectangles and the bag of words method is applied to each rectangle separately (Figure S7). The resulting feature vector is a concatenation of the 1+4+16 = 21 dictionaries. This approach has been shown to be highly successful in machine vision tasks[3,4]. The down side of this approach is that it inflates the feature dimensionality significantly, and requires reducing the dictionary size. In our experiments, we tested a dictionary size 100, which provides similar accuracies as the dictionary size of 500 used in the rest of the analysis (as shown above).

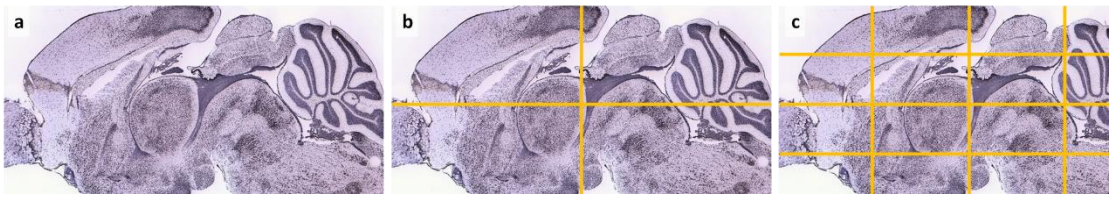


Figure S7: A spatial pyramid approach to extracting dense SIFT features. Features were extracted in the full image (a) and the image divided into four parts (b) and 16 parts (c).

The spatial pyramid approach yielded an overall mean AUC of 0.6231, which is slightly and insignificantly lower than the mean AUC obtained without the pyramidal kernel, 0.6322. We conclude that the increase in feature dimensionality hurts more than the gain obtained by describing different brain regions separately.

These results illustrate the challenging tradeoff when computing both local and global features. An alternative approach could be based on data-dependent segmentation of images into anatomic structures (like the thalamus, cortex or cerebellum) followed by coding each structure separately. Such segmentation is a topic for a separate research.

1. Lein, E. S. *et al.* Genome-wide atlas of gene expression in the adult mouse brain. *Nature* **445**, 168–76 (2007).

2. Lazebnik, S., Schmid, C. & Ponce, J. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Volume 2 CVPR06* **2**, 2169–2178 (2006).

3. Grauman, K. & Darrell, T. *The pyramid match kernel: discriminative classification with sets of image features*. *Tenth IEEE International Conference on Computer Vision ICCV05 Volume 1* **2**, 1458–1465 (Ieee: 2005).

4. Huang, T. Linear spatial pyramid matching using sparse coding for image classification. *2009 IEEE Conference on Computer Vision and Pattern Recognition* 1794–1801 (2009).doi:10.1109/CVPR.2009.5206757

2.4 Coordinate-descent for learning orthogonal matrices through Givens rotations

Coordinate-descent for learning orthogonal matrices through Givens rotations

Uri Shalit URI.SHALIT@MAIL.HUJI.AC.IL
ICNC-ELSC & Computer Science Department, The Hebrew University of Jerusalem, 91904 Jerusalem Israel
The Gonda Brain Research Center, Bar Ilan University, 52900 Ramat-Gan, Israel

Gal Chechik GAL.CHECHIK@BIU.AC.IL
The Gonda Brain Research Center, Bar Ilan University, 52900 Ramat-Gan, Israel

Abstract

Optimizing over the set of orthogonal matrices is a central component in problems like sparse-PCA or tensor decomposition. Unfortunately, such optimization is hard since simple operations on orthogonal matrices easily break orthogonality, and correcting orthogonality usually costs a large amount of computation.

Here we propose a framework for optimizing orthogonal matrices, that is the parallel of coordinate-descent in Euclidean spaces. It is based on *Givens-rotations*, a fast-to-compute operation that affects a small number of entries in the learned matrix, and preserves orthogonality.

We show two applications of this approach: an algorithm for tensor decompositions used in learning mixture models, and an algorithm for sparse-PCA. We study the parameter regime where a Givens rotation approach converges faster and achieves a superior model on a genome-wide brain-wide mRNA expression dataset.

1. Introduction

Optimization over orthogonal matrices – matrices whose rows and columns form an orthonormal basis of $\mathbb{R}^d$ – is central to many machine learning optimization problems. Prominent examples include *Principal Component Analysis* (PCA), *Sparse PCA*, and *Independent Component Analysis (ICA)*. In addition, many new applications of tensor orthogonal decompositions were introduced recently, including Gaussian Mixture Models, Multi-view Models and Latent Dirichlet Allocation (e.g., Anandkumar et al. (2012a); Hsu & Kakade (2013)).

Proceedings of the 31^{st} International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

A major challenge when optimizing over the set of orthogonal matrices is that simple updates such as matrix addition usually break orthonormality. Correcting by orthonormalizing a matrix $V \in \mathbb{R}^{d\times d}$ is typically a costly procedure: even a change to a single element of the matrix, may require $O(d^3)$ operations in the general case for re-orthogonalization.

In this paper, we present a new approach for optimization over the manifold of orthogonal matrices, that is based on a series of sparse and efficient-to-compute updates that operate **within the set of orthonormal matrices**, thus saving the need for costly orthonormalization. The approach can be seen as the equivalent of coordinate descent in the manifold of orthonormal matrices. Coordinate descent methods are particularly relevant for problems that are too big to fit in memory, for problems where one might be satisfied with a partial answer, or in problems where not all the data is available at one time (Richtárik & Takáč, 2012).

We start by showing that the orthogonal-matrix equivalent of a single coordinate update is applying a single *Givens rotation* to the matrix. In section 3 we prove that for a differentiable objective the procedure converges to a local optimum under minimal conditions, and prove an $O(1/T)$ convergence rate for the norm of the gradient. Sections 4 and 5 describe two applications: (1) sparse PCA, including a variant for streaming data; (2) a new method for orthogonal tensor decomposition. We study how the performance of the method depends on the problems hyperparameters using synthetic data, and demonstrate that it achieves superior accuracy on an application of sparse-PCA for analyzing gene expression data.

2. Coordinate descent on the orthogonal matrix manifold

Coordinate descent (CD) is an efficient alternative to gradient descent when the cost of computing and applying a gradient step at a single coordinate is small relative to com-

puting the full gradient. In these cases, convergence can be achieved with a smaller number of computing operations, although using a larger number of (faster) steps.

Applying coordinate descent to optimize a function involves choosing a coordinate basis, usually the standard basis. Then calculating a directional derivative in the direction of one of the coordinates. And finally, updating the iterate in the direction of the chosen coordinate. To generalize CD to operate over the set of orthogonal matrices, we need to generalize these ideas of directional derivatives and updating the orthogonal matrix in a "straight direction".

In the remaining of this section, we introduce the set of orthogonal matrices, $\mathcal{O}_d$, as a Riemannian manifold. We then show that applying coordinate descent to the Riemannian gradient amounts to multiplying by Givens rotations. Throughout this section and the next, the objective function is assumed to be a differentiable function $f : \mathcal{O}_d \to \mathbb{R}$.

2.1. The orthogonal manifold and Riemannian gradient

The orthogonal matrix manifold $\mathcal{O}_d$ is the set of $d \times d$ matrices U such that $UU^T = U^TU = I_d$. It is a $\frac{d(d+1)}{2}$ dimensional smooth manifold, and is an embedded submanifold of the Euclidean space $R^{d\times d}$ (Absil et al., 2009).

Each point $U \in \mathcal{O}_d$ has a tangent space associated with it, a $\frac{d(d-1)}{2}$ dimensional vector space, that we will use below in order to capture the notion of 'direction' on the manifold. The tangent space is denoted $T_U\mathcal{O}_d$, and defined by $T_U\mathcal{O}_d = \{Z \in \mathbb{R}^{d\times d}, Z = U\Omega : \Omega = -\Omega^T\} = USkew(d)$, where $Skew(d)$ is the set of skew-symmetric $d \times d$ matrices.

2.1.1. Geodesic directions

The natural generalization of straight lines to the manifold context are *geodesic curves*. A geodesic curve is locally the shortest curve between two points on the manifold, or equivalently, a curve with no acceleration tangent to the manifold (Absil et al., 2009). For a point $U \in \mathcal{O}_d$ and a "direction" $U\Omega \in T_U\mathcal{O}_d$ there exists a single geodesic line that passes through U in direction Ω. Fortunately, while computing a geodesic curve in the general case might be hard, computing it for $\mathcal{O}_d$ has a closed form expression: $\gamma : (-1, 1) \to \mathcal{O}_d$, $\gamma(\theta) = U\text{Expm}(\theta\Omega)$, where $\gamma(\theta)$ with $\theta \in (-1, 1)$ is the parameterization of the curve, and Expm is the matrix exponential function.

In the special case where the operator $Expm(\Omega)$ is applied to a skew-symmetric matrix Ω, it maps Ω into an orthogonal matrix [1]. As a result, $\gamma(\theta) = U\text{Expm}(\theta\Omega)$ is also an orthogonal matrix for all θ.

[1] Because $\text{Expm}(\Omega)\text{Expm}(\Omega)^T = \text{Expm}(\Omega)\text{Expm}(\Omega^T) = \text{Expm}(\Omega)\text{Expm}(-\Omega) = I$

2.1.2. The directional derivative

In analogy to the Euclidean case, the Riemannian directional derivative of f in the direction of a vector $U\Omega \in T_U\mathcal{O}_d$ is defined as the derivative of a single variable function which involves looking at f along a single curve (Absil et al., 2009):

$$\nabla_\Omega f(U) \equiv \frac{\mathrm{d}}{\mathrm{d}\theta} f(\gamma(\theta))\Big|_{\theta=0} = \frac{\mathrm{d}}{\mathrm{d}\theta} f(U\text{Expm}(\theta\Omega))\Big|_{\theta=0}. \tag{1}$$

Note that $\nabla_\Omega f(U)$ is a scalar. The definition means that the directional derivative is f' with f restricted to the geodesic curve going through U in the direction $U\Omega$.

2.1.3. The directional update

Since the Riemannian equivalent of walking in a straight line is walking along the geodesic curve, taking a step of size $\eta > 0$ from a point $U \in \mathcal{O}_d$ in direction $U\Omega \in T_U\mathcal{O}_d$ amounts to:

$$U_{next} = U\text{Expm}\left(\eta\Omega\right), \tag{2}$$

We also have to define the orthogonal basis for $Skew(d)$. Here we use $\{e_ie_j^T - e_je_i^T : 1 \le i < j \le d\}$. We denote each basis vector as $H_{ij} = e_ie_j^T - e_je_i^T$, $1 \le i < j \le d$.

2.2. Givens rotations as coordinate descent

Coordinate descent is a popular method of optimization in Euclidean spaces. It can be more efficient than computing full gradient steps when it is possible to (1) compute efficiently the coordinate directional derivative, and (2) apply the update efficiently. We will now show that in the case of the orthogonal manifold, applying the update (step 2) can be achieved efficiently. The cost of computing the coordinate derivative (step 1) depends on the specific nature of the objective function f, and we we show below several cases where that can be achieved efficiently.

Let H_{ij} be a coordinate direction, let $\nabla_{H_{ij}} f(U)$ be the corresponding directional derivative, and choose step size $\eta > 0$. A straightforward calculation based on Eq. 2 shows that the update $U_{next} = U\text{Expm}(-\eta H_{ij})$ obeys

$$\text{Expm}(-\eta H_{ij}) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & cos(\eta) & \cdots & -sin(\eta) & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & sin(\eta) & \cdots & cos(\eta) & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

This matrix is known as a *Givens rotation* (Golub & Van Loan, 2012) and is denoted $G(i,j,-\eta)$. It has $cos(\eta)$ at the (i,i) and (j,j) entries, and $\pm sin(\eta)$ at the (j,i) and (i,j) entries. It is a simple and sparse orthogonal matrix. For a dense matrix $A \in \mathbb{R}^{d\times d}$, the linear operation $A \mapsto AG(i,j,\eta)$ rotates the i^{th} and j^{th} columns of A by an angle η in the plane they span. Computing this operation costs $6d$ multiplications and additions. As a result, computing Givens rotations successively for all $\frac{d(d-1)}{2}$ coordinates H_{ij} takes $O(d^3)$ operations, the same order as ordinary matrix multiplication. Therefore the relation between the cost of a single Givens relative to a full gradient update is the same as the relation between the cost of a single coordinate update and a full update is in Euclidean space. We note that any determinant-1 orthogonal matrix can be decomposed into at most $\frac{d(d-1)}{2}$ Givens rotations.

2.3. The Givens rotation coordinate descent algorithm

Based on the definition of Givens rotation, a natural algorithm for optimizing over orthogonal matrices is to perform a sequence of rotations, where each rotation is equivalent to a coordinate-step in CD.

To fully specify the algorithm we need two more ingredients: (1) Selecting a schedule for going over the coordinates and (2) Selecting a step size. For scheduling, we chose here to use a random order of coordinates, following many recent coordinate descent papers (Richtárik & Takáč, 2012; Nesterov, 2012; Patrascu & Necoara, 2013).

For choosing the step size η we use exact minimization, since we found that for the problems we aim to solve, using exact minimization was usually the same order of complexity as performing approximate minimization (like using an Armijo step rule Bertsekas (1999); Absil et al. (2009)).

Based on these two decisions, Algorithm (1) is a random coordinate minimization technique.

Algorithm 1 Riemannian coordinate minimization on $\mathcal{O}_d$

Input: Differentiable objective function f, initial matrix $U_0 \in \mathcal{O}_d$
$t = 0$
while not converged **do**
1. Sample uniformly at random a pair $(i(t), j(t))$ such that $1 \le i(t) < j(t) \le d$.
2. $\theta_{t+1} = \underset{\theta}{\operatorname{argmin}}\, f\left(U_t \cdot G(i,j,\theta)\right)$.
3. $U_{t+1} = U_t \cdot G(i,j,\theta_{t+1})$.
4. $t = t+1$.
end while
Output: U_{final}.

3. Convergence rate for Givens coordinate minimization

In this section, we show that under the assumption that the objective function f is differentiable Algorithm 1 converges to critical point of the function f, and the only stable convergence points are local minima. We further show that the expectation w.r.t. the random choice of coordinates of the squared l_2-norm of the Riemannian gradient converges to 0 with a rate of $O(\frac{1}{T})$ where T is the number of iterations. The proofs, including some auxiliary lemmas, are provided in the supplemental material. Overall we provide the same convergence guarantees as provided in standard non-convex optimization (e.g., Nemirovski (1999); Bertsekas (1999)).

Definition 1. *Riemannian gradient*
The Riemannian gradient $\nabla f(U)$ of f at point $U \in \mathcal{O}_d$ is the matrix $U\Omega$, where $\Omega \in Skew(d)$, $\Omega_{ji} = -\Omega_{ij} = \nabla_{ij} f(U), 1 \le i < j \le d$ is defined to be the directional derivative as given in Eq. 1, and $\Omega_{ii} = 0$. The norm of the Riemannian gradient $||\nabla f(U)||^2 = Tr(\nabla f(U) \nabla f(U)^T) = ||\Omega||^2_{fro}$.

Definition 2. *A point $U_* \in \mathcal{O}_d$ is* asymptotically stable *with respect to Algorithm (1) if it has a neighborhood $\mathcal{V}$ of U_* such that all sequences generated by Algorithm (1) with starting point $U_0 \in \mathcal{V}$ converge to U_*.*

Theorem 1. *Convergence to local optimum*
(a) The sequence of iterates U_t of Algorithm (1) satisfies: $\lim_{t\to\infty} ||\nabla f(U_t)|| = 0$. This means that the accumulation points of the sequence $\{U_t\}_{t=1}^{\infty}$ are critical points of f.
(b) Assume the critical points of f are isolated. Let U_* be a critical point of f. Then U_* is a local minimum of f if and only if it is asymptotically stable with regard to the sequence generated by Algorithm (1).

Definition 3. *For an iteration t of Algorithm (1), and a set of indices $(i(t), j(t))$, we define the auxiliary single variable function g_t^{ij} :*

$$g_t^{ij}(\theta) = f\left(U_t \cdot G(i,j,\theta)\right), \tag{3}$$

Note that g_t^{ij} are differentiable and periodic with a period of 2π. Since $\mathcal{O}_d$ is compact and f is differentiable there exists a single Lipschitz constant $L(f) > 0$ for all g_t^{ij}.

Theorem 2. *Rate of convergence*
Let f be a continuous function with L-Lipschitz directional derivatives [2]. Let U_t be the sequence generated by Algorithm 1. For the sequence of Riemannian gradients $\nabla f(U_t) \in T_{U_t}\mathcal{O}_d$ we have:

$$\max_{0\le t\le T} E\left[||\nabla f(U_t)||_2^2\right] \le \frac{L \cdot d^2 \left(f(U_0) - f_{min}\right)}{T+1} \quad . \tag{4}$$

[2] Because $\mathcal{O}_d$ is compact, any function f with a continuous second-derivative will obey this condition.

The proof is a Riemannian version of the proof for the rate of convergence of Euclidean random coordinate descent for non-convex functions (Patrascu & Necoara, 2013) and is provided as supplemental material.

4. Sparse PCA

Principal component analysis (PCA) is a basic dimensionality reducing technique used throughout the sciences. Given a data set $A \in \mathbb{R}^{d\times n}$ of n observations in d dimensions, the principal components are a set of orthogonal vectors $z_1, z_2, \ldots, z_m \in \mathbb{R}^d$, such that the variance $\sum_{i=1}^m z_i^T AA^T z_i$ is maximized. The data is then represented in a new coordinate system $\hat{A} = Z^T A$ where $Z = [z_1, z_2, \ldots, z_m] \in \mathbb{R}^{d\times m}$.

One drawback of ordinary PCA is lack of interpretability. In the original data A, each dimension usually has an understandable meaning, such as the level of expression of a certain gene. The dimensions of $\hat{A}$ however are typically linear combinations of all gene expression levels, and as such are much more difficult to interpret. A common approach to the problem of finding *interpretable* principal components is Sparse PCA (Zou et al., 2006; Journée et al., 2010; d'Aspremont et al., 2007; Zhang et al., 2012; Zhang & Ghaoui, 2012). SPCA aims to find vectors z_i as in PCA, but which are also sparse. In the gene-expression example, the non-zero components of z_i might correspond to a few genes that explain well the structure of the data A.

One of the most popular approaches for solving the problem of finding sparse principal components is the work by Journée et al. (2010). In their paper, they formalize the problem as finding the optimum of the following constrained optimization problem to find the sparse basis vectors Z:

$$\underset{U\in\mathbb{R}^{n\times m}, Z\in\mathbb{R}^{d\times m}}{\operatorname{argmax}} \; Tr(Z^T AU) - \gamma \sum_{ij} |Z_{ij}| \qquad (5)$$

$$s.t.\;\; U^T U = I_m, \sum_{i=1}^{d} Z_{ij}^2 = 1 \;\forall j = 1 \ldots m \quad .$$

Journée et al. provide an algorithm to solve Eq. 5 that has two parts: The first and more time consuming part finds an optimal U, from which optimal Z is then found. We focus here on the problem of finding the matrix U. Note that when $m = n$, the constraint $U^T U = I_m$ implies that U is an orthogonal matrix.

We use a second formulation of the optimization problem, also given by Journée et al. in section 2.5.1 of their paper:

$$\underset{U\in\mathbb{R}^{n\times m}}{\operatorname{argmax}} \sum_{j=1}^{m}\sum_{i=1}^{d} [|(A\cdot U)_{ij}| - \gamma]_+^2$$
$$s.t.\;\; U^T U = I_m,$$

where n is the number of samples, d is the input dimensionality and m is the number of PCA components computed. This objective is once-differentiable and the objective matrix U grows with the number of samples n.

4.1. Givens rotation algorithm for the full case $m = n$

If we choose the number of principal components m to be equal to the number of samples n we can apply Algorithm ((1)) directly to solve the optimization problem of Eq. 6. Explicitly, at each round t, for choice of coordinates (i, j) and a matrix $U_t \in \mathcal{O}_d$, the resulting coordinate minimization problem is:

$$\underset{\theta}{\operatorname{argmin}} - \sum_{j=1}^{m}\sum_{i=1}^{d} [|(AU_t G(i,j,\theta))_{ij}| - \gamma]_+^2 =$$
$$\underset{\theta}{\operatorname{argmin}} - \sum_{k=1}^{d} [|cos(\theta)(AU_t)_{ki} + sin(\theta)(AU_t)_{kj}| - \gamma]_+^2 +$$
$$[| - sin(\theta)(AU_t)_{ki} + cos(\theta)(AU_t)_{kj}| - \gamma]_+^2 \qquad (6)$$

Algorithm 2 Riemannian coordinate minimization for sparse PCA

Input: Data matrix $A \in \mathbb{R}^{d\times n}$, initial matrix $U_0 \in \mathcal{O}_n$, sparsity parameter $\gamma \geq 0$
$t = 0$
$AU = A \cdot U_0$.
while not converged **do**
1. Sample uniformly at random a pair $(i(t), j(t))$ such that $1 \leq i(t) < j(t) \leq n$.
2. $\theta_{t+1} = \underset{\theta}{\operatorname{argmax}}$ $\sum_{k=1}^{d}([|cos(\theta)(AU)_{ki(t)} + sin(\theta)(AU)_{kj(t)}| - \gamma]_+^2$ $+[| - sin(\theta)(AU)_{ki(t)} + cos(\theta)(AU)_{kj(t)}| - \gamma]_+^2)$.
3. $AU = AU \cdot G(i(t), j(t)), \theta_{t+1})$.
4. $t = t + 1$.
end while
5. $Z = solveForZ(AU, \gamma)$ // Algorithm 6 of Journée et al. (2010).
Output: $Z \in \mathbb{R}^{d\times n}$

See Algorithm (2) for the full procedure. In practice, there is no need to store the matrices U_t in memory, and one can work directly with the matrix AU_t. Evaluating the expression in Eq. 6 for a given θ requires $O(d)$ operations, where d is the dimension of the data. We found in practice that optimizing Eq. 6 required an order of 5-10 evaluations. Overall each iteration of Algorithm (2) requires $O(d)$ operations.

4.2. Givens rotation algorithm for the case $m < n$

The major drawback of Algorithm (2) is that it requires the number of principal components m to be equal to the num-

ber of samples n. This kind of "full dimensional sparse PCA" may not be necessary when researchers are interested to obtain a small number of components. We therefore develop a streaming version of Algorithm (2). For a small given m, we treat the data as if only m samples exist at any time, giving an intermediate model $AU \in \mathbb{R}^{d\times m}$. After a few rounds of optimizing over this subset of samples, we use a heuristic to drop one of the previous samples and incorporate a new sample. This gives us a streaming version of the algorithm because in every phase we need only m samples of the data in memory. The full details of the algorithm are given in the supplemental material.

4.3. Experiments

Sparse PCA attempts to trade-off two variables: the fraction of data variance that is explained by the model's components, and the level of sparsity of the components. In our experiment, we monitor a third important parameter, the number of floating point operations (FLOPS) performed to achieve a certain solution. To compute the number of FLOPS we counted the number of additions and multiplications computed on each iteration. This does not include pointer arithmetic.

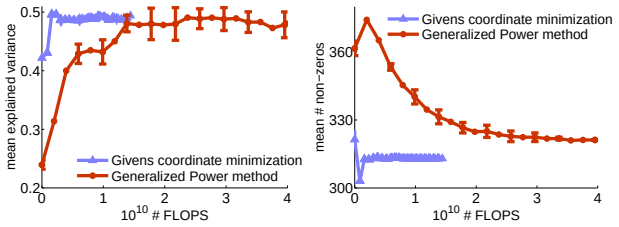
We first examined Algorithm 2 for the case where $m = n$. We used the prostate cancer gene expression data by Singh et al. (2002). This dataset consists of the gene expression levels for 52 tumor and 50 normal samples over 12,600 genes, resulting in a $12,600 \times 102$ data matrix.

We compared the performance of our approach with that of the *Generalized Power Method* of Journée et al. (2010). We focus on this method for comparisons because both methods optimize the same objective function, which allows to characterize the relative strengths and weaknesses of the two approaches.

As can be seen in Figure 1, the Givens coordinate minimization method finds a sparser solution with better explained variance, and does so faster than the generalized power method.

We tested the streaming version of the coordinate descent algorithm for sparse PCA (Algorithm 5, supp. material) on a recent large gene expression data set collected from of six human brains (Hawrylycz et al., 2012). Overall, each of the 20K human genes was measured at 3702 different brain locations, and this data can be used to study the spatial patterns of mRNA expression across the human brain. We again compared the performance of our approach with that of the *Generalized Power Method* of Journée et al. (2010).

We split the data into 5 train/test partitions, with each train set including 2962 examples and each test set including 740 examples. We evaluated the amount of variance explained by the model on the test set. We use the adjusted variance procedure suggested in this case by Zou et al. (2006), which takes into account the fact that the sparse principal components are not orthogonal.



(a) explained variance (b) number of non-zeros

Figure 1. (a) The explained variance as function of FLOPS of the coordinate minimization method from Algorithm 2 and of the generalized power method by Journée et al. (2010), on a prostate cancer gene expression dataset. (b) The number of non-zeros in the sparse PCA matrix as function of FLOPS of the coordinate minimization method from Algorithm 2 and of the generalized power method by Journée et al. (2010), on a prostate cancer gene expression dataset. The size of the sparse PCA matrix is $12,600 \times 102$.

For the Generalized Power Method we use the greedy l_1 version of Journée et al. (2010), with the parameter μ set to 1. We found the greedy version to be more stable and to be able to produce sparse solutions when the number of components was $m > 1$. We used values of γ ranging from 0.01 to 0.2, and two stopping conditions: "convergence", where the algorithm was run until its objective converged within a relative tolerance level of 10^{-4}, and "early stop" where we stopped the algorithm after 14% of the iterations required for convergence. For our algorithm we used the same range of γ values, and an early-stop condition where the algorithm was stopped after using 14% of the samples.

Figure 2 demonstrates the tradeoff between floating point operations and explained variance for SPCA with 3, 5 and 10 components and with 3 sparsity levels: 5%, 10% and 20%. Using low dimensions is often useful for visual exploration of the data. Each dot represents one instance of the algorithm, run with a certain value of γ and stopping criterion. To avoid clutter we only show instances which performed best in terms of explained variance or few FLOPS.

When strong sparsity is required (5% or 10% sparsity), the Givens-rotation coordinate descent algorithm finds solutions faster (blue rectangles are more to the left in Figure 2), and these solutions are similar or better in terms of explained variance. For low-dimensional less sparse solutions (20% sparsity) we find that the generalized power method finds comparable or better solutions using the same computational cost, but only when the number of components is small, as seen in Figure 2.c,f,i.

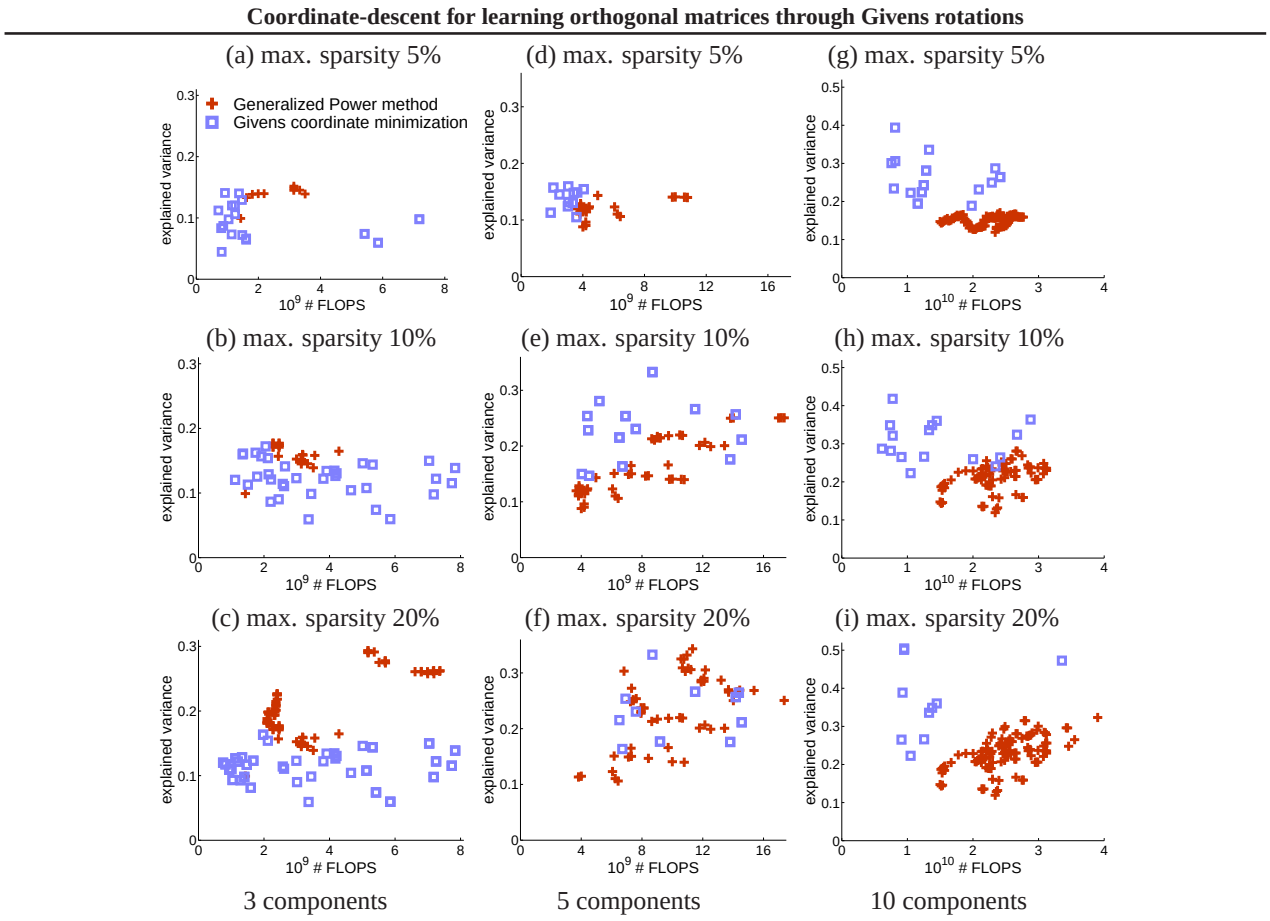


Figure 2. The tradeoff between explained variance and computational cost for 3, 5 and 10-component sparse PCA models applied to human gene expression data. The models are constrained for maximum sparsity of 5% (a), (d) & (g), 10% (b), (e) & (h) and 20% (c), (f) & (i). Red pluses indicate the Generalized Power method (Journée et al., 2010); blue squares represent the Givens coordinate procedure. See Subsection 4.3 for experimental conditions. Explained variance was adjusted following Zou et al. (2006).

5. Orthogonal tensor decomposition

Recently it has been shown that many classic machine learning problem such as Gaussian Mixture Models and Latent Dirichlet Allocation can be solved efficiently by using 3rd order moments (Anandkumar et al., 2012a; Hsu & Kakade, 2013; Anandkumar et al., 2012b;c; Chaganty & Liang, 2013). These methods ultimately rely on finding an orthogonal decomposition of 3-way tensors $T \in \mathbb{R}^{d\times d\times d}$, and reconstructing the solution from this decomposition. Here we show that the problem of finding an orthogonal decomposition for a tensor $T \in \mathbb{R}^{d\times d\times d}$ can be naturally cast as an optimization problem over the orthogonal matrix manifold. We apply Algorithm 1 to this problem, and compare its performance on a task of finding a Gaussian Mixture Model with a state-of-the-art tensor decomposition method, the robust Tensor Power Method (Anandkumar et al., 2012a). We find that the Givens coordinate method consistently finds better solutions when the number of mixture components is large.

5.1. Orthogonal tensor decomposition

The problem of tensor decomposition is very hard in general (Kolda & Bader, 2009). However, a certain class of tensors known as *orthogonally decomposable* tensors are easier to decompose, as has been discussed recently by Anandkumar et al. (2012a); Hsu & Kakade (2013) and others. Here we introduce the problem of orthogonal tensor decomposition, and provide a new characterization of the solutions to the decomposition problem as extrema of an optimization problem on the orthogonal matrix manifold.

The resulting algorithm is similar to one recently proposed by Ishteva et al. (2013). However, we aim for full diagonalization, while they focus on finding a good low-rank approximation. This results in different objective functions: ours involves third-order polynomials on $\mathcal{O}_d$, while Ishteva et al.'s results in sixth-order polynomials on the low-rank compact Stiefel manifold. Diagonalizing the tensor T is attainable in our case thanks to the strong assumption that

it is orthogonally decomposable. Nonetheless, both methods are extensions of Jacobi's eigenvalue algorithm to the tensor case, in different setups.

We start with preliminary notations and definitions. We focus here on symmetric tensors $T \in \mathbb{R}^{d\times d\times d}$. A third-order tensor is symmetric if its values are identical for any permutation σ of the indices: with $T_{i_1 i_2 i_3} = T_{i_{\sigma(1)} i_{\sigma(2)} i_{\sigma(3)}}$.

We also view a tensor T as a trilinear map. $T : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$: $T(v_1, v_2, v_3) = \sum_{a,b,c=1}^{d} T_{abc} v_{1a} v_{2b} v_{3c}$.

Finally, we also use the three-form tensor product of a vector $u \in \mathbb{R}^d$ with itself: $u \otimes u \otimes u \in \mathbb{R}^{d\times d\times d}$, $(u \otimes u \otimes u)_{abc} = u_a \cdot u_b \cdot u_c$. Such a tensor is called a *rank-one* tensor.

Definition 4. *A symmetric tensor T is orthogonally decomposable if there exists an orthonormal set $v_1, \dots v_d \in \mathbb{R}^d$, and positive scalars $\lambda_1, \dots \lambda_d > 0$ such that:*

$$T = \sum_{i=1}^{d} \lambda_i (v_i \otimes v_i \otimes v_i). \tag{7}$$

Unlike matrices, most symmetric tensors are not orthogonally decomposable. However, as shown by Anandkumar et al. (2012a); Hsu & Kakade (2013); Anandkumar et al. (2013), several problems of interest, notably Gaussian Mixture Models and Latent Dirichlet Allocation do give rise to third-order moments which are orthogonally decomposable in the limit of infinite data.

The goal of orthogonal tensor decomposition is, given an orthogonally decomposable tensor T, to find the orthogonal vector set $v_1, \dots v_d \in \mathbb{R}^d$ and the scalars $\lambda_1, \dots \lambda_d > 0$.

We now show that finding an orthogonal decomposition can be stated as an optimization problem over $\mathcal{O}_d$:

Theorem 3. *Let $T \in R^{d\times d\times d}$ have an orthogonal decomposition as in Definition 4, and consider the optimization problem*

$$\max_{U\in\mathcal{O}_d} f(U) = \sum_{i=1}^{d} T(u_i, u_i, u_i), \tag{8}$$

where $U = [u_1\ u_2\ \dots\ u_d]$. The stable stationary points of the problem are exactly orthogonal matrices U such that $u_i = v_{\pi(i)}$ for a permutation π on $[d]$. The maximum value they attain is $\sum_{i=1}^{d} \lambda_i$.

The proof is given in the supplemental material.

5.2. Coordinate minimization algorithm for orthogonal tensor decomposition

We now adapt Algorithm 1 for solving the problem of orthogonal tensor decomposition of a tensor T, by maximizing the objective function 8, $f(U) = \sum_{i=1}^{d} T(u_i, u_i, u_i)$. For this we need to calculate the form of the function $g_t^{ij}(\theta) = f\left(U \cdot G(i,j,\theta)\right)$. We have:

$$g_t^{ij}(\theta) = f\left(U \cdot G(i,j,\theta)\right) = \sum_{k\neq i,j}^{d} T(u_k, u_k, u_k) + T\left(\tilde{u}_i,\ \tilde{u}_i,\ \tilde{u}_i\right) + T\left(\tilde{u}_j,\ \tilde{u}_j,\ \tilde{u}_j\right).$$

where we used $\tilde{u}_i = cos(\theta)u_i + sin(\theta)u_j$ and $\tilde{u}_j = cos(\theta)u_j - sin(\theta)u_i$.

Denote by $\tilde{T}$ the tensor such that: $\tilde{T}_{ijk} = T(u_i, u_j, u_k)$. We will abuse notation and denote $\tilde{T} = T(U, U, U)$. The tensor $\tilde{T}$ is the three-way multiplication of T by the matrix U. This is the lifting of the matrix operation $\tilde{M} = M(U,U) = UMU^T$ to the tensor domain.

Collecting terms, using the symmetry of T and some basic trigonometric identities, we then have:

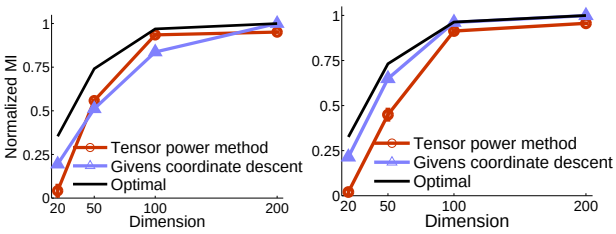
$$\begin{aligned} g_t^{ij}(\theta) =& cos^3(\theta)\left(\tilde{T}_{iii} + \tilde{T}_{jjj} - 3\tilde{T}_{ijj} - 3\tilde{T}_{jii}\right) \\ &+ sin^3(\theta)\left(\tilde{T}_{iii} - \tilde{T}_{jjj} - 3\tilde{T}_{ijj} + 3\tilde{T}_{jii}\right) \\ &+ cos(\theta)\left(3\tilde{T}_{ijj} + 3\tilde{T}_{jii}\right) \\ &+ sin(\theta)\left(3\tilde{T}_{ijj} - 3\tilde{T}_{jii}\right). \end{aligned} \tag{9}$$

In each step of the algorithm, we maximize $g_t^{ij}(\theta)$ over $-\pi \leq \theta < \pi$. The scalar function g_t^{ij} has at most 3 maxima that can be obtained in closed form solution, and thus can be maximized in constant time.

Algorithm 3 Riemannian coordinate maximization for orthogonal tensor decomposition

Input: Symmetric tensor $T \in \mathbb{R}^{d\times d\times d}$.
Initialize $t = 0$, $\tilde{T}^0 = T$, $U_0 = I_d$.
while not converged **do**
1. Sample uniformly at random a pair $(i(t), j(t))$ such that $1 \leq i(t) < j(t) \leq d$.
2. Obtain $\tilde{T}^t_{iii}$, $\tilde{T}^t_{jjj}$, $\tilde{T}^t_{ijj}$, $\tilde{T}^t_{jii}$.
3. $\theta_t = \underset{\theta}{\text{argmax}}\ g_t^{ij}(\theta)$, where g_t^{ij} is defined as in 9.
4. $\tilde{T}^{t+1} = \tilde{T}^t\left(G(i,j,\theta_t), G(i,j,\theta_t), G(i,j,\theta_t)\right)$.
// Three way multiplication of $\tilde{T}^t$ by $G(i,j,\theta_t)$.
5. $U_{t+1} = U_t G(i,j,\theta_t)$.
6. $t = t + 1$.
end while
Output: U_{final}.

The most computationally intensive part of Algorithm 3 is line 4. Multiplying a tensor by the Givens rotation $G(i,j,\theta)$ only affects tensor entries on the i-th and j-th slice. This requires $O(d^2)$ operations per iteration. In Section D of the supplemental material we provide a different version of this



(a) 10,000 samples (b) 200,000 samples

Figure 3. Clustering performance in terms of normalized MI of the Givens algorithm vs. the tensor power method of Anandkumar et al. (2012a). Clustering by fitting a GMM from samples drawn from a 20-component GMM with varying dimension, using 3rd order moments. Reconstruction is performed from (a) 10K and (b) 200K samples. Blue line with triangles marks the Givens coordinate method. Red line with circles marks the tensor power method, and the black line is the optimal performance if all GMM parameters are known.

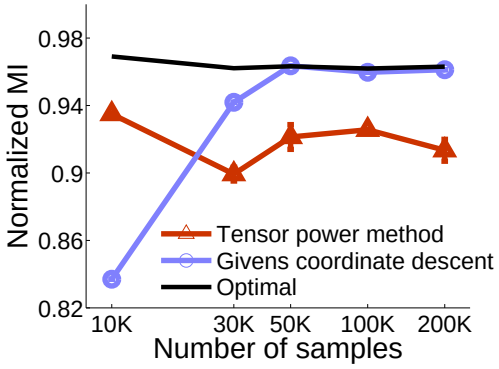


Figure 4. Same task as Figure 3, but for fixed dimension $d = 100$ and varying number of samples.

algorithm which does not require calculating the tensor T. Instead, it operates directly on the data points, calculating cross products on demand. This version of the algorithm has complexity per step of $O(\#samples)$ instead.

5.3. Experiments

Hsu & Kakade (2013) and Anandkumar et al. (2012a) have recently shown how fitting a Gaussian Mixture Model (GMM) with common spherical covariance can be reduced to orthogonally decomposing a third moment tensor. We evaluate the Givens coordinate minimization algorithm using this problem. We compare with a state of the art tensor decomposition method, the robust tensor power method, as given in Anandkumar et al. (2012a).

We generated GMMs with the following parameters: number of dimensions in $\{10, 20, 50, 100, 200\}$, number of samples in $\{10K, 30K, 50K, 100K, 200K\}$. We used 20 components, each with a spherical variance of 2. The centers were sampled from a Gaussian distribution with an inverse-Wishart distributed covariance matrix. Given the samples, we constructed the 3rd order moment, decomposed it, and reconstructed the model following the procedure in Anandkumar et al. (2012a). We then clustered the samples according to the reconstructed model, and measured the *normalized mutual information* (NMI) (Manning et al., 2008) between the learned clustering and the true clusters.

Figure 3 compares the performance of the two methods with the optimal NMI across dimensions. The coordinate minimization method outperforms the tensor power method for the large sample size (200K), whereas for small sample size (10K) the tensor power method performs better for the intermediate dimensions. In Figure 4 we see the performance of both algorithms across all sample sizes for dimension $= 100$. We see that the coordinate minimization method again performs better for larger sample sizes. We observed this phenomenon for 50 components as well, and for mixture models with larger variance.

6. Conclusion

We described a framework to efficiently optimize differentiable functions over the manifold of orthogonal matrices. The approach is based on Givens rotations, which we show can be viewed as the parallel of coordinate updates in Euclidean spaces. We prove the procedure's convergence to a local optimum. Using this framework, we developed algorithms for two unsupervised learning problems: Finding sparse principal components; and learning a Gaussian mixture model through orthogonal tensor decomposition. Our method poses an alternative to the tensor power method for orthogonal tensor decompositions. Our alternative extends the way the Jacobi eigenvalue algorithm is an alternative to the matrix power method for matrix decompositions.

We expect that the proposed framework can be further extended to other problems requiring learning over orthogonal matrices including ICA. Moreover, coordinate descent approaches have some inherent advantages and are sometimes better amenable to parallelization. Developing distributed Givens-rotation algorithms would be an interesting future research direction.

Acknowledgments

We wish to thank the anonymous reviewers for numerous improvements to the paper, and Haim Avron for fruitful discussions. U.S. is a recipient of the Google Europe Fellowship in Machine Learning, and this research is supported in part by this Google Fellowship. G.C. was supported by the Israeli science foundation grant 1090/12, and by a Marie Curie reintegration grant PIRG06-GA-2009-256566.

References

Absil, P-A, Mahony, Robert, and Sepulchre, Rodolphe. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

Anandkumar, Anima, Ge, Rong, Hsu, Daniel, Kakade, Sham M, and Telgarsky, Matus. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012a.

Anandkumar, Anima, Ge, Rong, Hsu, Daniel, and Kakade, Sham M. A tensor spectral approach to learning mixed membership community models. *arXiv preprint arXiv:1302.2684*, 2013.

Anandkumar, Animashree, Foster, Dean P, Hsu, Daniel, Kakade, Sham M, and Liu, Yi-Kai. A spectral algorithm for latent dirichlet allocation. *arXiv preprint arXiv:1204.6703*, 2012b.

Anandkumar, Animashree, Hsu, Daniel, and Kakade, Sham M. A method of moments for mixture models and hidden markov models. *arXiv preprint arXiv:1203.0683*, 2012c.

Bertsekas, Dimitri P. *Nonlinear programming*. Athena Scientific, 1999.

Chaganty, Arun Tejasvi and Liang, Percy. Spectral experts for estimating mixtures of linear regressions. *arXiv preprint arXiv:1306.3729*, 2013.

d'Aspremont, Alexandre, El Ghaoui, Laurent, Jordan, Michael I, and Lanckriet, Gert RG. A direct formulation for sparse pca using semidefinite programming. *SIAM review*, 49(3):434–448, 2007.

Golub, Gene H and Van Loan, Charles F. *Matrix computations*, volume 3. JHUP, 2012.

Hawrylycz, Michael J, Lein, S, Guillozet-Bongaarts, Angela L, Shen, Elaine H, Ng, Lydia, Miller, Jeremy A, van de Lagemaat, Louie N, Smith, Kimberly A, Ebbert, Amanda, Riley, Zackery L, et al. An anatomically comprehensive atlas of the adult human brain transcriptome. *Nature*, 489(7416):391–399, 2012.

Hsu, Daniel and Kakade, Sham M. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pp. 11–20. ACM, 2013.

Ishteva, Mariya, Absil, P-A, and Van Dooren, Paul. Jacobi algorithm for the best low multilinear rank approximation of symmetric tensors. *SIAM Journal on Matrix Analysis and Applications*, 34(2):651–672, 2013.

Journée, Michel, Nesterov, Yurii, Richtárik, Peter, and Sepulchre, Rodolphe. Generalized power method for sparse principal component analysis. *The Journal of Machine Learning Research*, 11:517–553, 2010.

Kolda, Tamara G and Bader, Brett W. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

Manning, Christopher D, Raghavan, Prabhakar, and Schütze, Hinrich. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

Nemirovski, A. Optmization II Numerical Methods for Nonlinear Continuous Optimization. 1999.

Nesterov, Yu. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Patrascu, Andrei and Necoara, Ion. Efficient random coordinate descent algorithms for large-scale structured nonconvex optimization. *arXiv preprint arXiv:1305.4027*, 2013.

Richtárik, Peter and Takáč, Martin. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, pp. 1–38, 2012.

Singh, Dinesh, Febbo, Phillip G, Ross, Kenneth, Jackson, Donald G, Manola, Judith, Ladd, Christine, Tamayo, Pablo, Renshaw, Andrew A, D'Amico, Anthony V, Richie, Jerome P, et al. Gene expression correlates of clinical prostate cancer behavior. *Cancer cell*, 1(2):203–209, 2002.

Zhang, Youwei and Ghaoui, Laurent El. Large-scale sparse principal component analysis with application to text data. *arXiv preprint arXiv:1210.7054*, 2012.

Zhang, Youwei, dAspremont, Alexandre, and El Ghaoui, Laurent. Sparse pca: Convex relaxations, algorithms and applications. In *Handbook on Semidefinite, Conic and Polynomial Optimization*, pp. 915–940. Springer, 2012.

Zou, Hui, Hastie, Trevor, and Tibshirani, Robert. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.

Supplemental material of Coordinate-descent for learning orthogonal matrices through Givens rotations

Uri Shalit URI.SHALIT@MAIL.HUJI.AC.IL
ICNC-ELSC & Computer Science Department, The Hebrew University of Jerusalem, 91904 Jerusalem Israel
The Gonda Brain Research Center, Bar Ilan University, 52900 Ramat-Gan, Israel

Gal Chechik GAL.CHECHIK@BIU.AC.IL
The Gonda Brain Research Center, Bar Ilan University, 52900 Ramat-Gan, Israel

A. Proofs of theorems of Section 3

Below we use a slightly modified definition of Algorithm 1. The difference lies only in the sampling procedure, and is essentially a technical difference to ensure that each coordinate step indeed improves the objective or lies at an optimum, so that the proofs could be stated more succinctly.

Algorithm 4 Riemannian coordinate minimization on $\mathcal{O}_d$, sampling variant

Input: Differentiable objective function f, initial matrix $U_0 \in \mathcal{O}_d$
$t = 0$
while not converged **do**
1. Sample coordinate pairs $(i(t), j(t))$ such that $1 \leq i(t) < j(t) \leq d$ uniformly at random without replacement, until the objective function can improve
2. $U_{t+1} = \underset{\theta}{\operatorname{argmin}}\ f\left(U_t \cdot G(i, j, \theta)\right)$.
3. $t = t + 1$.
end while

Definition 1. *A point $U_* \in \mathcal{O}_d$ is* asymptotically stable *with respect to Algorithm 4 if it has a neighborhood $\mathcal{V}$ of U_* such that all sequences generated by Algorithm 4 with starting point $U_0 \in \mathcal{V}$ converge to U_*.*

Theorem 1. Convergence to local optimum
(a) The sequence of iterates U_t of Algorithm 4 satisfies: $\lim_{t\to\infty} ||\nabla f(U_t)|| = 0$. This means that the accumulation points of the sequence $\{U_t\}_{t=1}^{\infty}$ are critical points of f.
(b) Assume the critical points of f are isolated. Let U_* be a critical point of f. Then U_* is a local minimum of f if and only if it is asymptotically stable with regard to the sequence generated by Algorithm 4.

Proof. (a) Algorithm 4 is obtained by taking a step in each iteration t in the direction of the tangent vector Z_t, such that for the coordinates $(i(t), j(t))$ we have $(Z_t)_{ij} = -(\nabla f(U_t))_{ij}$, $(Z_t)_{ji} = -(\nabla f(U_t))_{ji}$, and $(Z_t)_{kl} = 0$ for all other coordinates (k, l).

The sequence of tangent vectors $Z_t \in T_{U_t}\mathcal{O}_d$ is easily seen to be gradient related: $\limsup k \to \infty \langle \nabla f(U_t), Z_t\rangle < 0$ [1]. This follows from Z_t being equal to exactly two coordinates of $\nabla f(U_t)$, with all other coordinates being 0.

Using the optimal step size as we do assures at least as large an increase $f(U_t) - f(U_{t+1})$ as using the Armijo step size rule (Armijo, 1966; Bertsekas, 1999). Using the fact that the manifold $\mathcal{O}_d$ is compact, we obtain by theorem 4.3.1 and corollary 4.3.2 of Absil et al. (2009) that $\lim_{t\to\infty} ||\nabla f(U_t)|| = 0$

(b) Since Algorithm 4 produces a monotonically decreasing sequence $f(U_t)$, and since the manifold $\mathcal{O}_d$ is compact, we are in the conditions of Theorems 4.4.1 and 4.4.2 of Absil et al. (2009). These imply that the only critical points which are local minima are asymptotically stable.

□

We now provide a rate of convergence proof. This proof is a Riemannian version of the proof for the rate of convergence of Euclidean random coordinate descent for non-convex functions given by Patrascu & Necoara (2013).

Definition 2. *For an iteration t of Algorithm 4, and a set of indices $(i(t), j(t))$, we define the auxiliary single variable function g_t^{ij} :*

$$g_t^{ij}(\theta) = f\left(U_t \cdot G(i, j, \theta)\right), \tag{1}$$

Note that g_t^{ij} are differentiable and periodic with a period of 2π. Since $\mathcal{O}_d$ is compact and f is differentiable there exists a single Lipschitz constant $L(f) > 0$ for all g_t^{ij}.

[1] To obtain a rigorous proof we slightly complicated the sampling procedure in line 1 of Algorithm 1, such that coordinates with 0 gradient are not resampled until a non-zero gradient is sampled.

Theorem 2. Rate of convergence
Let f be a continuous function with L-Lipschitz directional derivatives [2]. Let U_t be the sequence generated by Algorithm 4. For the sequence of Riemannian gradients $\nabla f(U_t) \in T_{U_t}\mathcal{O}_d$ we have:

$$\max_{0\le t\le T} E\left[||\nabla f(U_t)||_2^2\right] \le \frac{L\cdot d^2\left(f(U_0)-f_{min}\right)}{T+1} \quad . \quad (2)$$

Lemma 1. *Let $g:\mathbb{R}\to\mathbb{R}$ be a periodic differentiable function, with period 2π, and $L-$Lipschitz derivative g'. Then there for all $\theta\in[-\pi\ \pi]$: $g(\theta)\le g(0)+\theta g'(0)+\frac{L}{2}\theta^2$.*

Proof. We have for all θ,
$|g'(\theta)-g'(0)|\le L|\theta|$. We now have: $g(\theta)-g(0)-\theta g'(0)=\int_0^\theta g'(\tau)-g'(0)d\tau\le\int_0^\theta|g'(\tau)-g'(0)|d\tau\le\int_0^\theta L|\tau|d\tau=\frac{L}{2}\theta^2$. □

Corollary 1. *Let $g=g^{t+1}_{i(t+1)j(t+1)}$. Under the conditions of Algorithm 4, we have:*
$f(U_t)-f(U_{t+1})\ge\frac{1}{2L}\nabla_{ij}f(U_t)^2$ for the same constant L defined in 1.

Proof. By the definition of g we have $f(U_{t+1})=\min_\theta g(\theta)$, and we also have $g(0)=f(U_t)$. Finally, by Eq. 1 of the main paper we have $\nabla_{ij}f(U_t)=g'(0)$. From Lemma 1, we have $g(\theta)-g(0)\le\theta g'(0)+\frac{L}{2}\theta^2$. Minimizing the right-hand side with respect to θ, we see that $\min_\theta\{g(0)-g(\theta)\}\ge\frac{1}{2L}(g'(0))^2$. Substituting $f(U_{t+1})=\min_\theta g(\theta)$,$f(U_t)=g(0)$, and $\frac{1}{2L}\nabla_{ij}f(U_t)=g'(0)$ completes the result. □

Proof of Theorem 2. By Corollary 1, we have $f(U_t)-f(U_{t+1})\ge\frac{1}{2L}\nabla_{ij}f(U_t)^2$. Recall that $\pm\nabla_{ij}f(U_t)$ is the (i,j) and (j,i) entry of $\nabla f(U_t)$. If we take the expectation of both sides with respect to a uniform random choice of indices i,j such that $1\le i<j\le d$, we have:

$$E\left[f(U_t)-f(U_{t+1})\right]\ge\frac{1}{L\cdot d^2)}||\nabla f(U_t)||^2, \quad (3)$$

Summing the left-hand side gives a telescopic sum which can be bounded by $f(U_0)-\min_{U\in\mathcal{O}_d}f(U)=f(U_0)-f_{min}$. Summing the right-hand side and using this bound, we obtain

$$\sum_{t=0}^{T}E\left[||\nabla f(U_t)||_2^2\right]\le L\cdot d^2(f(U_0)-f_{min}) \quad (4)$$

This means that $\min_{0\le t\le T}E\left[||\nabla f(U_t)||_2^2\right]\le\frac{L\cdot d^2(f(U_0)-f_{min})}{T+1}$. □

[2] Because $\mathcal{O}_d$ is compact, any function f with a continuous second-derivative will obey this condition.

B. Proofs of theorems of Section 5

Definition 4. A tensor T is *orthogonally decomposable* if there exists an orthonormal set of vectors $v_1,\dots v_d\in\mathbb{R}^d$, and positive scalars $\lambda_1,\dots\lambda_d>0$ such that:

$$T=\sum_{i=1}^{d}\lambda_i(v_i\otimes v_i\otimes v_i). \quad (5)$$

Theorem 3. Let $T\in R^{d\times d\times d}$ have an orthogonal decomposition as in Definition 4, and consider the optimization problem

$$\max_{U\in\mathcal{O}_d}f(U)=\sum_{i=1}^{d}T(u_i,u_i,u_i), \quad (6)$$

where $U=[u_1\ u_2\ \dots\ u_d]$. The stable stationary points of the problem are exactly orthogonal matrices U such that $u_i=v_{\pi(i)}$ for a permutation π on $[d]$. The maximum value they attain is $\sum_{i=1}^d\lambda_i$.

Proof. For a tensor T' denote $\text{vec}(T')\in\mathbb{R}^{d^3}$ the vectorization of T' using some fixed order of indices. Set $\hat{T}(U)=\sum_{i=1}^d(u_i\otimes u_i\otimes u_i)$, with $\hat{T}(U)_{abc}=\sum_{i=1}^d u_{ia}u_{ib}u_{ic}$. The sum of trilinear forms in Eq. 6 is equivalent to the inner product in $\mathbb{R}^{d^3}$ between $\hat{T}(U)$ and T: $\sum_{i=1}^d T(u_i,u_i,u_i)=\sum_{i=1}^d\sum_{abc}T_{abc}u_{ia}u_{ib}u_{ic}=\sum_{abc}T_{abc}\left(\sum_{i=1}^d u_{ia}u_{ib}u_{ic}\right)=\sum_{abc}T_{abc}\hat{T}(U)_{abc}=\text{vec}(T)\cdot\text{vec}(\hat{T}(U))$. Consider the following two facts:
(1) $\hat{T}(U)_{abc}\le 1\ \forall a,b,c=1\dots d$: since the vectors u_i are orthogonal, all their components $u_{ia}\le 1$. Thus $\hat{T}(U)_{abc}=\sum_{i=1}^d u_{ia}u_{ib}u_{ic}\le\sum_{i=1}^d u_{ia}u_{ib}=\le 1$, where the last inequality is because the sum is the inner product of two rows of an orthogonal matrix.
(2) $||\text{vec}(\hat{T}(U))||_2^2=d$. This is easily checked by forming out the sum of squares explicitly, using the orthonormality of the rows and columns of the matrix U.
Assume without loss of generality that $V=I_d$. This is because we may replace the terms $T(u_i,u_i,u_i)$ in the objective with $T(V^Tu_i,V^Tu_i,V^Tu_i)$, and because the manifold $V^T\mathcal{O}_d$ is identical to $\mathcal{O}_d$. Thus we have that T is a diagonal tensor, with $T_{aaa}=\lambda_a>0$, $a=1\dots d$. Considering facts (1) and (2) above, we have the following inequality:

$$\max_{U\in\mathcal{O}_d}\sum_{i=1}^{d}T(u_i,u_i,u_i)=\max_{U\in\mathcal{O}_d}\text{vec}(\hat{T}(U))\cdot T\le \quad (7)$$

$$\max_{\hat{T}}\text{vec}(\hat{T})\cdot T\quad s.t.\quad ||\text{vec}(\hat{T})||_\infty\le 1\wedge||\text{vec}(\hat{T})||_2^2=d. \quad (8)$$

T is diagonal by assumption, with exactly d non-zero entries. Thus the maximum of (5) is attained if and only if

Algorithm 5 Riemannian coordinate minimization for streaming sparse PCA

Input: Data stream $a_i \in \mathbb{R}^d$, number of sparse principal components m, initial matrix $U_0 \in \mathcal{O}_m$, sparsity parameter $\gamma \geq 0$, number of inner iterations L.
$AU = [a_1 a_2 \dots a_m] \cdot U_0$. //AU is of size $d \times m$
while not stopped **do**
  for $t = 1 \dots L$ **do**
    1. Sample uniformly at random a pair $(i(t), j(t))$ such that $1 \leq i(t) < j(t) \leq m$.
    2. $\theta_{t+1} = \operatorname{argmax}_{\theta} \sum_{k=1}^{d} (|cos(\theta)(AU)_{ki(t)} + sin(\theta)(AU)_{kj(t)}| - \gamma]_+^2 + [|-sin(\theta)(AU)_{ki(t)} + cos(\theta)(AU)_{kj(t)}| - \gamma]_+^2)$.
    3. $AU = AU \cdot G(i(t), j(t)), \theta_{t+1})$.
  end for
  4. $i_{min} = \operatorname{argmin}_{i=1\dots m} ||(AU)_{:,i}||_2$.
  5. Sample new data point a_{new}.
  6. $(AU)_{:,i_{min}} = a_{new}$.
end while
$Z = solveForZ(AU, \gamma)$ // Algorithm 6 of Journée et al. (2010).
Output: $Z \in \mathbb{R}^{d \times m}$

$\hat{T}_{aaa} = 1$, $a = 1 \dots d$, and all other entries of $\hat{T}$ are 0. The value at the maximum is then $\sum_{i=1}^{d} \lambda_i$.

The diagonal ones tensor $\hat{T}$ can be decomposed into $\sum_{i=1}^{d} e_i \otimes e_i \otimes e_i$. Interestingly, in the tensor case, unlike in the matrix case, the decomposition of orthogonal tensors is *unique* upto permutation of the factors (Kruskal, 1977; Kolda & Bader, 2009). Thus, the only solutions which attain the maximum of 7 are those where $u_i = e_{\pi(i)}$, $i = 1, \dots d$. □

C. Algorithm for streaming sparse PCA

Following are the details for the streaming sparse PCA version of our algorithm used in the experiments of section 4. The algorithm itself is brought in Algorithm 5. The algorithm starts with running the original coordinate minimization procedure on the first m samples. It then chooses the column with the least l_2 and replaces it with a new data sample, and then re-optimizes on the new set of samples. There is no need for it to converge in the inner iterations, and in practice we found that order m steps after each new sample are enough for good results.

D. Alternate version of orthogonal tensor decomposition algorithm - lazy tensor evaluation

Algorithm 3 in the main text is "Riemannian coordinate maximization for orthogonal tensor decomposition". The version presented there assumes that the full $d \times d \times d$ tensor T is given as input to the algorithm. Typically in the applications we consider here, this tensor is formed as a third order moment from a given dataset. Let $A \in \mathbb{R}^{d \times n}$ be the data matrix, consisting of n observation with d dimensions. In the simplest case we will have that $T_{ijk} = \sum_{l=1}^{n} A_{il} A_{jl} A_{kl}$. More complex cases (for example when applying the method to fit an LDA model) still require simple vector operations which cost $O(n)$ computations to obtain each value T_{ijk}.

We can therefore adopt a lazy computation model, and refrain from constructing the entire moment tensor T in advance. Instead we may calculate the entries T_{ijk} only on demand, and on each step apply the Givens rotation to the *data matrix* instead of the tensor. This requires $O(n)$ operations, as we will be rotating the i and j dimensions (rows) of the data matrix A. See Algorithm 6 below.

Overall the computational cost of each step of this version of the algorithm is $O(n)$ where n is the number of data samples. This is compared to $O(d^2)$ operations for the version presented in the main text, where d is typically not the original data dimension, but the number of latent variables such as latent topics in LDA or mixture components in a GMM. See Anandkumar et al. (2012) for more details.

Algorithm 6 Riemannian coordinate maximization for orthogonal tensor decomposition with lazy tensor evaluation

Input: Data matrix $A \in \mathbb{R}^{d \times n}$. Procedure $\mathcal{S}(A)$ for obtaining single tensor entries from A with computational cost $O(n)$.
Initialize $t = 0$, $\tilde{A}_0 = A$, $U_0 = I_d$.
while not converged **do**
  1. Sample uniformly at random a pair $(i(t), j(t))$ such that $1 \leq i(t) < j(t) \leq d$.
  2. Obtain T_{iii}, T_{jjj}, T_{ijj}, T_{jii} from $\tilde{A}_t$ by $\mathcal{S}(\tilde{A}_t)$.
  3. $\theta_t = \operatorname{argmax}_{\theta} g_t^{ij}(\theta)$, where g_t^{ij} is defined as in Eq. 9 of the main text.
  4. $\tilde{A}_{t+1} = G(i, j, \theta_t)^T \tilde{A}_t$.
  5. $U_{t+1} = U_t G(i, j, \theta_t)$.
  6. $t = t + 1$.
end while
Output: U_{final}.

References

Absil, P-A, Mahony, Robert, and Sepulchre, Rodolphe. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

Anandkumar, Anima, Ge, Rong, Hsu, Daniel, Kakade, Sham M, and Telgarsky, Matus. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012.

Armijo, Larry. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 16(1):1–3, 1966.

Bertsekas, Dimitri P. *Nonlinear programming*. Athena Scientific, 1999.

Journée, Michel, Nesterov, Yurii, Richtárik, Peter, and Sepulchre, Rodolphe. Generalized power method for sparse principal component analysis. *The Journal of Machine Learning Research*, 11:517–553, 2010.

Kolda, Tamara G and Bader, Brett W. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

Kruskal, Joseph B. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95 – 138, 1977. ISSN 0024-3795. doi: 10.1016/0024-3795(77)90069-6.

Patrascu, Andrei and Necoara, Ion. Efficient random coordinate descent algorithms for large-scale structured nonconvex optimization. *arXiv preprint arXiv:1305.4027*, 2013.

Chapter 3

Discussion

3.1 Summary of contributions of this thesis

In this thesis we presented methods for learning dyadic relationships, focusing on methods for large-scale and high-dimensional data. We first presented OASIS, a fast and effective streaming similarity learning algorithm, which at the time of publication achieved state-of-the-art results on the task of ranking images by similarity. OASIS is based on sparse updates to a full-rank matrix, and scales well with many samples. However, OASIS suffered from two potential drawbacks. First is that OASIS memory and computation complexity scales quadratically with the problem dimension. The second drawback is that there is no efficient way to enforce a positive semi-definite constraint on the matrix model learned by OASIS, hampering the ability to turn OASIS into a true metric, or deriving a proper embedding from it.

In Chapter 2.2 we introduced LORETA, a streaming algorithm for learning low-rank matrices. LORETA is a Riemannian stochastic gradient descent algorithm. Using the manifold structure allows for memory and computational efficiency, scaling linearly with the problem dimensionality and with the model rank. Furthermore, optimizing within the low-rank manifold is a key factor in avoiding the numerical instability often associated with learning low-rank models. This instability is especially problematic in the streaming setting because of its noisy nature. We applied LORETA both to similarity learning problems and to the task of learning a multi-label model for images. It has also been used by Lim and Lanckriet (2014) for ranking songs by similarity.

In Chapter 2.3 we took on a challenge presented by scientific applications of similarity learning. When similarity learning is applied to image or song retrieval, the main goal is to mimic human perceptions of similarity. On the other hand, in scientific applications such as brain imaging and genetics, scientists are interested in gaining insight about questions such as why two brain scans or two genes are similar. We tackled this challenge in the field of gene expression images of the brain. Our approach is easily generalized to other domains where rich knowledge-bases exist, such as proteomics or MRI scans.

In order to learn interpretable similarity models we first mapped the brain gene expression images into a high-dimensional semantic representation. We did this by learning multiple classifiers which mapped the gene expression images into semantically meaningful categories, taken from the Gene Ontology. The categories themselves have a directed acyclic graph structure relating general and specific categories. We then used the semantic meaning of the categories and the structure between the categories to obtain an interpretable similarity measure. Our method can explain why two gene expression images are similar by using semantic categories from the Gene Ontology, as well as distinguish between different types of similarities based on those same categories.

Finally, in Chapter 2.4 we took on the challenge of scaling up the learning of orthogonal matrices. We were motivated by the new applications of orthogonal matrices in tensor decomposition

for method-of-moments estimations of latent variable models (Anandkumar et al., 2012a,b; Hsu and Kakade, 2013; Anandkumar et al., 2014), as well as by applications to sparse PCA and independent component analysis. We used the manifold structure of the orthogonal matrix group to define the Riemannian equivalent of Euclidean coordinates. The equivalent turns out to be simple, sparse orthogonal matrices called Givens rotations, which are rotation matrices that fix an $(n-2)$-dimensional space and perform a rotation on the remaining 2-dimensional subspace. Any $n \times n$ orthogonal matrix can be decomposed into a product of $\frac{n(n-1)}{2}$ Givens rotations. We prove that successively applying Givens rotations to an orthogonal matrix is exactly the Riemannian equivalent of Euclidean coordinate updates. Thus our algorithm enjoys the same advantages of scalability, parallelization and speed of Euclidean coordinate methods (Nesterov, 2012; Richtárik and Takáč, 2014), while preserving the orthogonal structure at all steps. We showed that while our method is not faster compared with the state-of-the-art method for orthogonal tensor decomposition, our method is significantly more robust to noise. For sparse PCA, we showed our method is both faster and achieves better solutions for this highly non-convex problem.

In the future work section below we discuss some interesting questions that turned up during our work on learning orthogonal matrices, as well as some future applications.

3.2 Future work

To conclude this thesis we look into several possible future directions stemming from the work presented herein.

Hidden convexity on the orthogonal matrix manifold

When applying the Givens orthogonal coordinate descent algorithm presented in Chapter 2.4 to tensor decomposition, we noticed an intriguing phenomenon: the iterates seemed to consistently have a linear convergence rate to an optimal solution, as shown for example in Figure 3.1. This occurred in a wide variety of experimental conditions with both real and synthetic data, while our theoretical guarantees only provided for local optima, with a sublinear convergence rate.

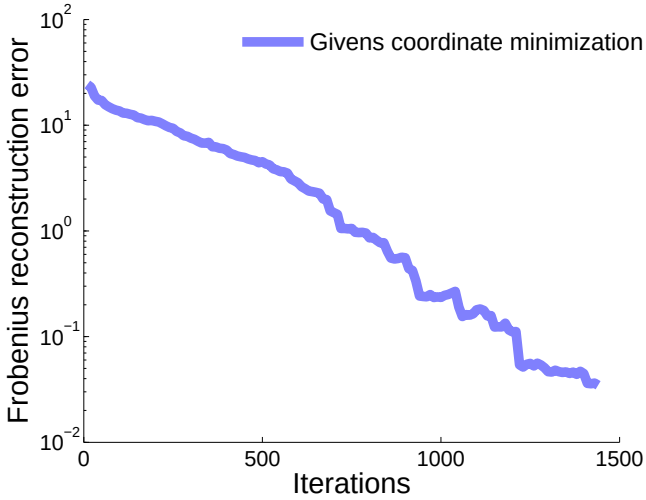


Figure 3.1: Convergence of the Riemannian coordinate descent algorithm presented in Chapter 2.4. Shown is the convergence of the 100 dimension model presented in Figure 3.b. of Chapter 2.4, where a Gaussian Mixture Model is fit using the the third order moment (following Anandkumar et al. (2012a)). A linear convergence rate is observed.

.

A recent paper by Belkin et al. (2014) gives a very interesting perspective which might explain this phenomenon. In the terms of Riemannian optimization, Belkin et al. introduce a quotient

structure on the orthogonal matrix manifold. They use the fact that in many cases, orthogonal matrix models are invariant to permutation and change of sign - this is for example the case in eigendecompositions. Thus, Belkin et al. "quotient out" this invariance. They then present a class of functions that are convex on that quotient structure, a class which includes the objective function for orthogonal tensor decomposition. The algorithm Belkin et al. employ is a form of Riemannian gradient descent. They show linear and super-linear convergence results to global optima. A natural question is how can the orthogonal coordinate methods presented in Chapter 2.4 be adapted to the quotient structure Belkin et al. propose, and how can the strong convergence theory be applied to orthogonal coordinate updates with Givens rotations.

Learning orthogonal matrices for matrix completion

Matrix completion (MC) is a well known problem in machine learning and data analysis (Candès and Recht, 2009; Candès and Tao, 2010; Keshavan et al., 2010). The problem is as follows: Assume there exists an approximately low-rank matrix $R \in \mathbb{R}^{n\times m}$, which we can only observe a small subset of its entries. The goal of MC is to reconstruct the matrix R from the subset of observed entries. A major motivation and application for MC is its importance in collaborative filtering for recommender systems, where the goal is recommending new items to a user based on the preferences of other users.

The usual formulation of the collaborative filtering approach to recommendation systems, is having input consisting of a set of triplets (i, j, r). This triplet indicates that user i has a preference r regarding item j. The preference r is a numeric value such as a rating from 1 to 5, or a binary value such a click or its absence. The task is to predict what will be user i's preference for an item j', for which she has not given a preference indication. The assumption is that such prediction is possible by using the preferences of other users who have in turn given feedback on item j', as well as other items. A standard mathematical formulation of the MC problem is as follows (Koren et al., 2009; Jain et al., 2013). Assume there are n users and m items, and a sparsely observed ratings matrix $R \in \mathbb{R}^{n\times m}$ as above. Denote by Ω the set of pairs (i, j) for which feedback exists. Denote by P_Ω the matrix projecting onto this set, such that $[P_\Omega(X)]_{ij} = X_{ij}$ if $(i, j) \in \Omega$, and $[P_\Omega(X)]_{ij} = 0$ otherwise. The goal is to find a *low-rank* factorization of R into a "user" matrix $A \in \mathbb{R}^{n\times k}$ and an "item" matrix $B \in \mathbb{R}^{m\times k}$, that has low mean squared error:

$$\min_{A,B} \|P_\Omega(R) - P_\Omega(AB^T)\|_F^2 = \min_{A,B} \sum_{j=1}^{m} \sum_{i|(i,j)\in\Omega} (A_i^T B_j - R_{ij})^2. \tag{3.1}$$

If one fixes the factor A in Eq. 3.1 above, then solving for B is a straightforward least-squares problem, and likewise if B is fixed and we are solving for A. Solving these two least-squares problems alternately until convergence is one of the standard algorithms for solving the matrix completion problem (Koren et al., 2009; Jain et al., 2013; Hardt, 2013). It is straightforward to show (Jain et al., 2013; Hardt, 2013) that in fact one can use an $n \times k$ orthogonal matrix U such that U spans the column space of A, instead of A itself.

Given such a matrix $U \in \mathbb{R}^{n\times k}$ spanning the column space of the user matrix A, the least square solution for each row $B_j \in \mathbb{R}^k$ of B has a closed form. Let $s(j) \subset [1 \dots n]$ denote the subset of users that have given a rating to the item j. Let $U_{s(j)} \in \mathbb{R}^{|s(j)|\times k}$ be the submatrix of the rows of U corresponding to those users. For simplicity we will assume that $U_{s(j)}$ is of rank-k, though this may be relaxed. Let $r_j \in \mathbb{R}^{|s(j)|}$ denote the ratings given by these $|s(j)|$ users to the item j. Then we have:

$$B_j = \left(U_{s(j)}^T U_{s(j)}\right)^{-1} U_{s(j)}^T r_j. \tag{3.2}$$

Substituting 3.2 into the objective 3.1 we obtain the following optimization problem:

$$\min_{U \in \mathbb{R}^{n \times k}} \sum_{j=1}^{m} \|U_{s(j)} \left(U_{s(j)}^T U_{s(j)}\right)^{-1} U_{s(j)}^T r_j - r_j\|_2^2 \tag{3.3}$$
$$s.t. \quad U^T U = I_k$$

Posing the problem of matrix completion as an optimization problem over orthogonal matrices lets one use efficient optimization tools such as Givens rotations and Householder reflections to find the optimal matrix subspace. This is further motivated by recent breakthroughs in the theoretical analysis of alternating minimization presented in the works of Jain et al. (2013), Hardt (2013) and Hardt and Wootters (2014). These papers all show that under certain conditions, alternating least-squares leads to an optimal solution of the MC problem while being computationally efficient. Interestingly, they all measure convergence in terms of the principal angles between the model and the optimal subspace. Thus this line of work is essentially about analyzing the properties of the orthogonal matrices spanning the "user" and "item" matrices. Therefore we believe that an interesting avenue for future research is understanding how minimizing Eq. 3.3 by means of Givens rotations or Householder reflections behaves both in terms of the principal angle to the optimal solution and in terms of computational complexity.

Efficient block-coordinate optimization for positive definite matrices

Our work on Givens rotations in Chapter 2.4 presented a way to perform low-complexity updates to a matrix while conserving a non-trivial global constraint - orthogonality. We are currently looking into a conceptually similar problem regarding positive definite (PD) matrices. As we have previously seen, PD matrices are useful as models for metric or similarity learning. Therefore, we wish to find a way to perform efficient block-coordinate updates to a PD matrix while conserving the PD constraint. Our motivation is the fact that the most common method of enforcing the PD constraint is computationally expensive - a full eigendecomposition. In Chapter 2.2 we dealt with this challenge in the case of low-rank positive semidefinite matrices by using the Riemannian manifold structure of the low-rank PD set. A drawback of the low-rank Riemannian manifold algorithm is its lack of convexity, making theoretical analysis very difficult.

A promising approach is learning a full-rank PD matrix while performing updates to a single column and row of the matrix in such a way that the PD constraint is automatically maintained. This has the added advantage of maintaining a PD model all along the optimization process, which is useful in the streaming setting.

The way we maintain the PD property is by using the Schur complement (Zhang, 2006) of the matrix model: Let A be an $n \times n$ PD matrix, and suppose without loss of generality that we wish to update the n-th column and n-th row of A (note that in order to maintain symmetry we must update both). Partition A as: $\begin{bmatrix} B & v^T \\ v & s \end{bmatrix}$ where B is the $(n-1) \times (n-1)$ upper-left block, v is an $n-1$ dimensional column vector, and s is a scalar. Our block-coordinate update involves updating v and s while keeping B fixed. By the Schur complement condition, the matrix A is PD if and only if the block B is PD and $s - v^T B^{-1} v > 0$. Since A is PD, we immediately have that B is PD as well, leaving us with a single quadratic constraint on v and s:

$$s - v^T B^{-1} v > 0. \tag{3.4}$$

Given an objective function f we minimize f as a function of $v \in \mathbb{R}^{n-1}$ and $s \in \mathbb{R}$ under the convex constraint 3.4. The minimization can be either exact or approximate, depending on the nature of the objective f. Note that the constraint 3.4 involves the inverse of the $(n-1) \times (n-1)$ block

B. This can be obtained efficiently by maintaining the inverse of A and performing a Sherman-Morrison-Woodbury matrix inverse update.

As is normally the case for coordinate optimization (Richtárik and Takáč, 2014; Nesterov, 2012), performing the update efficiently requires that f decomposes over the block we are using (a single row and column). Luckily, this is the case for a wide range of objectives, including those normally used for similarity learning such as the triplet loss we used for OASIS, and the loss function of the popular metric learning method LMNN (Weinberger et al., 2005; Weinberger and Saul, 2009).

Preliminary results indicate that the block-coordinate updates work at least as well as comparable metric and similarity learning methods. It does so while maintaining a PD model all along the optimization path, without resorting to an eigendecomposition.

New applications for interpretable similarity learning

The work presented in Chapter 2.3 of this thesis was focused on a specific application: Assisting scientists in understanding and exploring the complex relations between genes, their expression patterns in the brain, and their biological functions.

However, we believe that the ideas in Chapter 2.3 can be generalized to other domains and can be improved upon. One very interesting application is in the field of medical imaging such as MRI and X-ray scans. While there are quite a few attempts of applying similarity and metric learning to bioimaging (e.g. Godil et al., 2008; Shedden et al., 2009; Wei et al., 2009; Wernick et al., 2010; Chi et al., 2013), none have looked into the challenge of *explaining* the similarity predicted by the model. We believe that identifying similar scans while providing semantic interpretations is key in encouraging adoption in the medical community, while opening up new avenues for improving the model and its interaction with human experts. In addition, an interpretable similarity tool could be valuable in the training of health-care workers, as well as for the patients themselves. A relevant knowledge base for this application is the text and diagnoses given by doctors to these scans. These can be used as labels and featured as semantic interpretation that the model learns, similar to the role the Gene Ontology categories played in our work on gene expression images.

Bibliography

Pierre-Antoine Absil and Jérôme Malick. Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization*, 22(1):135–158, 2012.

Pierre-Antoine Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2009.

Yonatan Amit, Michael Fink, Nathan Srebro, and Shimon Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th International Conference on Machine Learning*, pages 17–24. ACM, 2007.

Anima Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012a.

Anima Anandkumar, Rong Ge, Daniel Hsu, and Sham M Kakade. A tensor spectral approach to learning mixed membership community models. *arXiv preprint arXiv:1302.2684*, 2013.

Animashree Anandkumar, Dean P Foster, Daniel Hsu, Sham M Kakade, and Yi-Kai Liu. A spectral algorithm for latent dirichlet allocation. *arXiv preprint arXiv:1204.6703*, 2012b.

Animashree Anandkumar, Rong Ge, Daniel Hsu, and Sham M Kakade. A tensor approach to learning mixed membership community models. *The Journal of Machine Learning Research*, 15 (1):2239–2312, 2014.

Aharon Bar-Hillel, Tomer Hertz, Noam Shental, and Daphna Weinshall. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6(6):937–965, 2005.

Mikhail Belkin, Luis Rademacher, and James Voss. Learning a hidden basis through imperfect measurements: An algorithmic primitive. *arXiv preprint arXiv:1411.1420*, 2014.

Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *Automatic Control, IEEE Transactions on*, 58(9):2217–2229, 2013.

Léon Bottou. Online learning and stochastic approximations. *On-line Learning in Neural Networks*, 17:9, 1998.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

Olivier Bousquet and Léon Bottou. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems*, pages 161–168, 2008.

Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *Information Theory, IEEE Transactions on*, 56(5):2053–2080, 2010.

Yanling Chi, Jiayin Zhou, Sudhakar K Venkatesh, Qi Tian, and Jimin Liu. Content-based image retrieval of multiphase ct images for focal liver lesion characterization. *Medical Physics*, 40(10): 103502, 2013.

Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *The Journal of Machine Learning Research*, 3:951–991, 2003.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7:551–585, 2006.

Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th International Conference on Machine Learning*, pages 209–216. ACM, 2007.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

Jia Deng, Alexander C Berg, and Li Fei-Fei. Hierarchical semantic indexing for large scale image retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 785–792. IEEE, 2011.

Manfredo P. Do Carmo. *Riemannian Geometry*. Birkhauser, 1992. ISBN 0817634908.

Susan T Dumais. Latent semantic analysis. *Annual Review of Information Science and Technology*, 38(1):188–230, 2004.

Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

Olivier Fercoq and Peter Richtárik. Accelerated, parallel and proximal coordinate descent. *arXiv preprint arXiv:1312.5799*, 2013.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1, 2010.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in Neural Information Processing Systems*, pages 2121–2129, 2013.

Gene Ontology Consortium. The gene ontology (go) database and informatics resource. *Nucleic Acids Research*, 32(suppl 1):D258–D261, 2004.

Afzal Godil, Benny Cheung, Asim Wagan, and Xiaolan Li. Bio-imaging toolkit for indexing, searching, navigation, discovery and annotation. In *Advances in Visual Computing*, pages 915–923. Springer, 2008.

Gene H Golub and Charles F Van Loan. *Matrix Computations*, volume 3. JHUP, 2012.

Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

Matthieu Guillaumin, Thomas Mensink, Jakob Verbeek, and Cordelia Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 309–316. IEEE, 2009a.

Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Is that you? metric learning approaches for face identification. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 498–505. IEEE, 2009b.

Moritz Hardt. Understanding alternating minimization for matrix completion. *arXiv preprint arXiv:1312.0925*, 2013.

Moritz Hardt and Mary Wootters. Fast matrix completion without the condition number. In *Proceedings of The 27th Conference on Learning Theory*, pages 638–678, 2014.

Michael Hawrylycz, Lydia Ng, David Feng, Susan Sunkin, Aaron Szafer, and Chinh Dang. The allen brain atlas. In *Springer Handbook of Bio-/Neuroinformatics*, pages 1111–1126. Springer, 2014.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear svm. In *Proceedings of the 25th International Conference on Machine Learning*, pages 408–415. ACM, 2008.

Daniel Hsu and Sham M Kakade. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, pages 11–20. ACM, 2013.

Prateek Jain, Brian Kulis, and Kristen Grauman. Fast image search for learned metrics. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

Prateek Jain, Raghu Meka, and Inderjit S Dhillon. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*, pages 937–945, 2010.

Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proc. STOC*, pages 665–674. ACM, 2013.

Thorsten Joachims. Making large scale svm learning practical. 1999.

Michel Journée, Yurii Nesterov, Peter Richtárik, and Rodolphe Sepulchre. Generalized power method for sparse principal component analysis. *The Journal of Machine Learning Research*, 11:517–553, 2010.

Sham M Kakade, Shai Shalev-Shwartz, and Ambuj Tewari. Efficient bandit algorithms for online multiclass prediction. In *Proceedings of the 25th International Conference on Machine Learning*, pages 440–447. ACM, 2008.

Raghunandan Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from noisy entries. In *Advances in Neural Information Processing Systems*, pages 952–960, 2009.

Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *Information Theory, IEEE Transactions on*, 56(6):2980–2998, 2010.

Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer, IEEE*, 42(8):30–37, 2009.

Brian Kulis. Metric learning: A survey. *Foundations & Trends in Machine Learning*, 5(4):287–364, 2012.

Ed S Lein, Michael J Hawrylycz, Nancy Ao, Mikael Ayres, Amy Bensinger, Amy Bernard, Andrew F Boe, Mark S Boguski, Kevin S Brockway, Emi J Byrnes, et al. Genome-wide atlas of gene expression in the adult mouse brain. *Nature*, 445(7124):168–176, 2006.

Li-Jia Li, Hao Su, Li Fei-Fei, and Eric P Xing. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Advances in Neural Information Processing Systems*, pages 1378–1386, 2010.

Daryl Lim and Gert Lanckriet. Efficient learning of mahalanobis metrics for ranking. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1980–1988, 2014.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction To Information Retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995.

Deanna Needell, Nathan Srebro, and Rachel Ward. Stochastic gradient descent and the randomized kaczmarz algorithm. *arXiv preprint arXiv:1310.5715*, 2013.

Yu Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Lydia Ng, Amy Bernard, Chris Lau, Caroline C Overly, Hong-Wei Dong, Chihchau Kuan, Sayan Pathak, Susan M Sunkin, Chinh Dang, Jason W Bohland, et al. An anatomic gene expression atlas of the adult mouse brain. *Nature Neuroscience*, 12(3):356–362, 2009.

Qi Qian, Rong Jin, Jinfeng Yi, Lijun Zhang, and Shenghuo Zhu. Efficient distance metric learning by adaptive sampling and mini-batch stochastic gradient descent (sgd). *arXiv preprint arXiv:1304.1192*, 2013.

Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. *arXiv preprint arXiv:1109.5647*, 2011.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 693–701, 2011.

Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *arXiv preprint arXiv:1212.0873*, 2012.

Peter Richtárik and Martin Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1-2):1–38, 2014.

Shai Shalev-Shwartz and Ambuj Tewari. Stochastic methods for l 1-regularized loss minimization. *The Journal of Machine Learning Research*, 12:1865–1892, 2011.

Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. *arXiv preprint arXiv:1212.1824*, 2012.

Kerby Shedden, Qian Li, Fangyi Liu, Young Tae Chang, and Gus R Rosania. Machine vision-assisted analysis of structure-localization relationships in a combinatorial library of prospective bioimaging probes. *Cytometry Part A*, 75(6):482–493, 2009.

Martin Takáč, Avleen Bijral, Peter Richtárik, and Nathan Srebro. Mini-batch primal and dual methods for svms. *arXiv preprint arXiv:1303.2314*, 2013.

Lorenzo Torresani, Martin Szummer, and Andrew Fitzgibbon. Efficient object category recognition using classemes. In *Computer Vision–ECCV 2010*, pages 776–789. Springer, 2010.

Du Tran and Alexander Sorokin. Human activity recognition with metric learning. In *Computer Vision–ECCV 2008*, pages 548–561. Springer, 2008.

Pavan Turaga, Ashok Veeraraghavan, Anuj Srivastava, and Rama Chellappa. Statistical computations on grassmann and stiefel manifolds for image and video-based recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(11):2273–2286, 2011.

Bart Vandereycken, Pierre-Antoine Absil, and Stefan Vandewalle. Embedded geometry of the set of symmetric positive semidefinite matrices of fixed rank. In *Statistical Signal Processing, 2009. SSP'09. IEEE/SP 15th Workshop on*, pages 389–392. IEEE, 2009.

Liyang Wei, Yongyi Yang, Miles N Wernick, and Robert M Nishikawa. Learning of perceptual similarity from expert readers for mammogram retrieval. *Selected Topics in Signal Processing, IEEE Journal of*, 3(1):53–61, 2009.

Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009.

Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 1473–1480, 2005.

Miles N Wernick, Yongyi Yang, Jovan G Brankov, Grigori Yourganov, and Stephen C Strother. Machine learning in medical imaging. *Signal Processing Magazine, IEEE*, 27(4):25–38, 2010.

Pengcheng Wu, Steven CH Hoi, Hao Xia, Peilin Zhao, Dayong Wang, and Chunyan Miao. Online multimodal deep similarity learning with application to image retrieval. In *Proceedings of the 21st ACM International Conference on Multimedia*, pages 153–162. ACM, 2013.

Tong Tong Wu and Kenneth Lange. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, pages 224–244, 2008.

Eric P Xing, Michael I Jordan, Stuart Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems*, pages 505–512, 2002.

Fuzhen Zhang. *The Schur Complement and its Applications*, volume 4. Springer, 2006.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of The Twenty-First International Conference on Machine Learning*, page 116. ACM, 2004.

Sheng-hua Zhong, Yan Liu, and Yang Liu. Bilinear deep learning for image classification. In *Proceedings of the 19th ACM International Conference on Multimedia*, pages 343–352. ACM, 2011.

תקציר

רבות מבעיות למידת-המכונה המודרניות נסובות על למידת מיפוי בין שני תחומים בעלי מימד גבוה. למשל, למידת מיפוי בין תמונות למילים, או למידה של אילו מסמכים דומים זה לזה. אנו קוראים למיפויים אלה *יחסים דיאדיים*. בתזה זו אנו מתמקדים בלמידת יחסים דיאדיים המקודדים בצורת מטריצה. לדוגמה, מטריצה הממפה בין ייצוג של תמונה לבין ייצוג סמנטי, או מטריצה הפועלת כתבנית בילינארית המקודדת דימיון בין מסמכים.

באופן ספציפי, אנו מתמקדים באתגר של למידת יחסים דיאדיים עבור מאגרי נתונים גדולים ובעלי מימד גבוה. אנו עושים שימוש בשני סטים של כלים על מנת לפתח את השיטות שיוצגו להלן: ראשית, אנו משתמשים במבנה הגאומטרי והאלגברי העשיר של מטריצות. בכלל זה, מבנה היריעה הרימני של קבוצת המטריצות נמוכות הדרגה, פירוקים של מטריצות אורתוגונליות, ותכונות של מטריצות מוגדרות חיובית. בנוסף, אנו מתרכזים באלגוריתמים של מידע זורם. אלה אלגוריתמים אשר בכל עת הינם בעלי גישה לתת-קבוצה מצומצמת של דגימות או מאפיינים, מתוך כלל מאגר המידע. אלגוריתמים של מידע זורם הינם המפתח להתמודדות עם מאגרי נתונים ענקיים אשר לא ניתן לטעון לזיכרון RAM, ובנוסף מאפשרים ניבוי מהיר עוד בטרם כלל מאגר המידע זמין או מעובד.

תזה זו מבוססת על ארבעה מאמרים. הראשון עוסק בבעיה של למידת מידת דימיון בין דוגמאות רב-מימדיות דלילות, עם יישומים בתחום של אחזור תמונות. אנו מתמקדים בתכנון אלגוריתם מהיר, המותאם למאגרי נתונים גדולים, וחוקרים כיצד אילוצי סימטריה וחיוביות משפיעים על ביצועיו. המאמר השני מציג שני אלגוריתמים של מידע זורם ללמידת מטריצות נמוכות-דרגה ומטריצות נמוכות-דרגה מוגדרות חיובית. אנו משתמשים בו ללמידת מודלים של דימיון ומודלים של תיוג מרובה תוויות. האלגוריתמים מבוססים על צעדי גרדיאנט סטוכסטיים במרחב רימני, ומציגים הן מהירות והן יציבות למרות האופי הלא-קמור של אילוץ הדרגה הנמוכה. המאמר השלישי עוסק באתגר של למידת יחסים הניתנים לפירוש – האם אלגוריתם למידת-מכונה יכול להסביר את הפלט שלו במונחים שיהיו מובנים והגיוניים עבור בן-אדם? המוטיבציה לעבודה זו באה מן הצורך להבין מאגר נתונים גדול של תמונות, המייצגות ברזולוציה גבוהה את רמת הביטוי של 20,000 גנים במח העכבר. בתמונות קיימות תבניות מורכבות בסקאלות שונות, ובין הגנים עצמם קיימת פעילות גומלין רבת פנים. שני גורמים אלה תורמים להפיכת מאגר נתונים זה לקשה להבנה ולניתוח. השיטה שאנו מציגים משתמשת בידע סמנטי שנאסף ממאגרי ידע עשירים המכילים מידע אודות גנים ותפקודם הביולוגי. שיטתנו מעניקה למדענים אמצעים לחקור ולהבין בצורה מעמיקה יותר את מאגר הנתונים. ולבסוף, המאמר הרביעי עוסק בבעיה הכללית של למידה יעילה של מטריצות אורתוגונליות. עניינו בבעייה זו התעורר בעקבות יישומים חדשים של מטריצות אורתוגונליות בפירוק טנזורים, עבור שערוך מודלים של משתנים חבויים באמצעות שיטת המומנטים. אנו מציגים שיטה חדשה לעדכון קואורדינטות רימני, המבוססת על מטריצות אורתוגונליות פשוטות ודלילות בשם "סיבובי גיבנס" (Givens Rotations). אנו מראים כי בהשוואה לשיטה המובילה כיום, שיטתנו מביאה לפתרונות טובים יותר של בעית פירוק הטנזורים. כמו כן שיטתנו מהירה ומביאה לפתרונות טובים יותר בהשוואה לשיטה מובילה בבעיית sparse PCA.

לסיום, אנו דנים במספר כיווני מחקר עתידיים הנובעים מהעבודה המוצגת כאן.

עבודה זו נעשתה בהדרכתם של:

פרופסור דפנה ויינשל

פרופסור גל צ'צ'ייק

למידת יחסים דיאדיים
ממידע זורם

חיבור לשם קבלת תואר דוקטור לפילוסופיה

מאת

אורי שליט

הוגש לסנט האוניברסיטה העברית בירושלים

ינואר 2015