Implications of Multimodal Deep Learning for Textual and Visual Data

Hieu Pham
Stanford University
Stanford, CA 94305
hyhieu@cs.stanford.edu

Abstract

While deep learning has been successful in a wide range of tasks in different fields, most of the current neural network based systems are still learning in merely one source of knowledge such as text, image or wave. In this work, we propose a novel method that acquires knowledge from both textual and visual data. Our analyses also show that it was possible to learn a function that map an arbitrary image or sequence of words into vectors in a joint high dimensional semantic space. These vectors are then evaluated on two vision-cross-language tasks; image-caption relevance ranking and image retrieval from text query. Different from prior work in the same direction, we only have a decent model on the text data and a weak model on image data. Furthermore, both the quantity and quality of our training text corpus are far better than those of our image dataset. From the poor performance of our vision model, we achieve $15\times$ *improvement after incorporating the knowledge from our textual models.*

1. Introduction

Knowledge in the real world exists in various domains. While our human brains can operate consistently across many of these domains, for example, when we see the word "*dog*" we think of the small animal with four legs, it is not trivial for machine learning algorithms to unify them. On the other hand, being able to acquire multimodal knowledge is very valuable. For instance, as shown in [32] while more data are being available, they do not have the same potential to be passed into machine learning algorithms. Specifically, some learning algorithms, usually computer vision ones, require that the training data is annotated by human experts, which are time consuming and expensive; while other data such as text corpora can be directly trained on with minimal preprocessing. Multimodal learning allows the latter class of algorithms to aid the former one with their knowledge on the same subject but from different source.

In this work, we propose a novel approach to connect knowledge from two sources: text and image. Our models on natural language are very strong, are trained on prosperous data and have been proved to achieve decent performances on several tasks [16, 28]. On the other hand, our visual model is week and also, we do not have a lot of good data to train it. We show that training the visual model alone is insufficient for several visions-cross-language tasks. However, by simply aiding the weak visual model with decent pretrained knowledge learned by the textual model, the joint model can capture important semantic information from images. We investigate our proposal on two tasks: image-caption relevance ranking and image retrieval from text query. In these tasks, our joint model performs $15\times$ better than the image model on its own. While we do not achieve state of the art performance on the aforementioned tasks, our work shows the potential of incorporating textual knowledge into vision problems.

The rest of this paper is structured as followed. In *Section 2*, we give the background of our work and revise the previous success in the direction. Next, in *Section 3*, we present our method. In *Section 4*, we describe our data sets, our experiments as well as our training procedures. Then, in *Section 5*, we discuss several aspects of our model and suggest improvements for subsequent work. Finally, *Section 6* is our conclusion.

2. Related Work

Our work draws inspiration from the general framework called representation learning. In this framework, the goal is to learn a map to represent input data, such as signals, images or texts, as high dimensional vectors in a semantic space. These high dimensional vectors are known as embeddings or distributed representations of input data. Our work focus on the embeddings of image and text data. Prior to our work, there have been ideas to learn such representations in for each domain as well as for multiple domains.

The latter situation is referred to as multimodal learning [25]. We now briefly revise the background work in each of the directions.

2.1. Sentence embedding

Deep learning methods have been successful in a wide range of tasks in natural language processing [21, 7, 4, 5]. A common trend among these methods is the presence of word embeddings, which overcomes the sparsity of natural language by representing each word in the vocabulary as a high dimensional vector. These embeddings then become the crucial components of a neural network, which is trained to achieve an objective function. The objective function can be either supervised as in [27], or pseudo-supervised as in [1, 24, 30], where pseudo-supervised means that there is no expert annotated training data, but the objectives are drawn from the nature of languages. Word embeddings are the excerpts from the larger models learned by these methods.

The generalization from embeddings of words into those of phrases and sentences is not trivial. The direct reason is that one cannot store one vector for each phrase or sentence of a language as one does for words. Thus a compositional model is needed to compute the representations of phrases and sentences from smaller linguistic components. Syntax-based compostional models such as [26, 27] recursively compute phrases and sentences embeddings based on precomputed constituency parse tree. Simpler methods such as [8, 13] simply takes the weighted sum of vectors of the words or bi-grams in the phrases or sentences. Closely related to our work is a new idea proposed by [16]. We will discuss this work in more details when we present it as a component of our method.

2.2. Image embedding

Image embeddings are the similar idea for words, phrases and sentences but applied on images. However, naturally, images are associated to neither recursive tree models nor sequential structures like phrases or sentences. It is not even feasible to define the "smallest units" of images as to words, the counterpart of language. Therefore, different approaches have been proposed such as convnet [19, 18] or autoencoder [2]. In [19], a new type of neural network layer, called the convolutiona layer, is introduced with the ability to share parameters along the layer to featurize the input. Similar to normal neural network layers, convolutional layers are followed by activation layers such as [9, 20] make the model non-linear, and then probably pool layers to sample into smaller size. Similar to NLP neural models, convnet is asked to give predictions specific to a problem, and then the predicting error will be backpropagated to train the parameters of the whole network. One can think of the final output of a convnet as the embedding for image.

In the previous paragraph, to obtain the prediction errors, it requires a human expert to annotate the images, usually by classification or by giving a caption. There are alternated unsupervised approach to compute image embedding, among which the most closely related idea to our work is autoencoder [10, 2]. In this approach, an input image is passed through several neural network layers, such as Restricted Boltzmann Machine [15] or convolutional [22] to downsize to into a fixed length vector. This vector is then asked to predict exactly the input image. This process may happen in several steps, where at each step, one can make the network output the error discrepancy from a desired reconstruction objective. The network parameters are learned by backpropagating these errors, and the last hidden layer of the network that is not asked to give reconstruct information is treated as the image embedding. The intuition behind autoencoder is that if a smaller dimensional vector can predict information in the image, it must carry important semantics of the image.

2.3. Joint semantic embedding

Given that one can learn embeddings for texts and images, a natural question is whether the two are relevant. This issue has been addressed in [32], where the distributed representations of images and their captions are mapped into a joint high dimension semantic space, and the similarity between two representations are measured by their scalar product, where larger is more similar. A more recent work in the same direction is [6], where the authors pretrain their word embeddings and sentence model and then use a large margin-based loss objective to learn the embedding vectors for images. The results of both work are functions that map images and texts into the same semantic space, where inputs with similar meanings have their targets in the space close to each other, according to their corresponding metrics. These functions can be used, for example, as a measure of similarity for the image-caption relevance ranking task as well as the image retrieval task.

This is also our objective. However, unlike [32], where the authors have to restrict their distributed representations to a boxed region, otherwise the scalar product can be scaled to arbitrarily large, we use the L_2 distance between the embedding, which is more intuitive. Also, unlike [6], where similarity objective is measured as a learnable quadratic form on the embedding vectors of image and text, our L_2 distance is a less general. Furthermore, compare to the work of these authors, our work uses a much weaker vision model, yet strong textual model. Our reasoning is that there are more text corpora that better models can be trained unsupervised on, so we attempt using their decent result to guide the training process of visual data, which are rarer and usually more expensive to collect.

3. Methods

In this section, we describe the methods used in this work. Our objective is to map images and their captions into a joint semantic space. This is a high dimensional vector space, in which representatives of objects and phenomena described by either visual or textual data are vectors, which are also referred to as embeddings or distributed representations. We want the vectors to satisfy, to an extent that we will quantify later, the constraint that similar objects and phenomena are represented by vectors that are close to each other in the space. Here, the metric for closeness is simply the L_2 distance. We start by describing several approaches we map sentences, and more generally, any sequence of words, into the semantic space. These embeddings are learned independently from the map for images. Next, we describe our approach to map images into the same semantic space with guiding signals from the captions of that image.

3.1. Sentence embedding

To learn the distributed representations for sentences, we start from those of words. We train *word2vec* by [23] on an English text corpus to precompute the embeddings of all English words. As a result, we obtain for each word a D-dimensional vector. These word vectors demonstrate many interesting semantic implications, for example the vectorial difference $king - queen$ and that of $man - woman$ are roughly the same, i.e. they have small L_2 distance from each other. To extend word embeddings to sentence embeddings, we need to use a compostional model. In this work, we use the two following ones

Average bag of words (ABOW) is the method where we take the average of the embeddings of all the words that appear in the sentence to be the representation of that sentence. This is a simple and time-efficient approach. However, there are certainly a lot of setbacks for ABOW, among which the most crucial is perhaps that ABOW loses all information about word order. For example, two sentences "*a dog is chasing a cat*" and "*a cat is chasing a dog*" are represented by the same vector in our semantic space.

Paragraph vector (PV), proposed by [16] is much more sophisticated. The method is based on the assumption that the representation vector of a sentence, which captures its semantic, should be capable of predicting the words that appear in the sentence. Given the assumption, a model is trained to minimize the prediction error on the N-grams in the training data. At test time, when the model sees an unseen sentence, a sentence vector is initialized, then trained using the same method used at train time until it converges to a point, which is treated as the sentence vector. PV successfully embeds important semantic information, such as sentiment, of sentences, and hence we expect that they give good representation for the image captions, which is our main objective. The PV model itself has a set of word vectors trained with the model from scratch. In this work, we use the word vectors computed by *word2vec* as the initialization for PV and hence we just need to train other parameters. One of the concerns about PV that is not addressed in [16] is the possibility that when PV sees two new sentences, in theory, the model can return two different vectorial representation because of the random initialization. This possibility poses a problem because we later want to use these sentence vectors to guide the learning of process of our images. However, in one of our unpublished work [**?**], we have proved that the objective function of PV is convex. This convexity implies that one global optimal sentence vector exists, and thus training methods such as gradient descent should give a sentence vector close to that theoretical vector, hence eases the concern of nondeterministic sentence vectors.

Having the two compostional models, we can ignore their details and instead, just think of them as black boxes that output gold sentence embeddings to guide the process of learning image embeddings.

3.2. Autoencoder

Figure 1 illustrates our architecture for image embedding. The key technique we use to compute image embedding is a deep convolutional neural network. At train time, given an image with arbitrary size, we first sample it into a fixed size input. This input is then passed to our neural network, which consists of two convolutional-rectified linear-pool combinations mounted in series. For simplicity, we always use 3×3 size for all receptive fields and pool size of 2. Furthermore, for reasons that will be clear later, we adjust the architecture of our neural network such that the result of the second pool layer has exactly D neurons. This constraint leads to the fixed number of receptive fields our second convolutional layer can have, which is $\frac{D}{W/2 \cdot H/2}$, where W, H are the width and height of the input layer to the network. On the other hand, we allow the first convolutional layer to have any number of receptive fields.

The output of the second pool layer is straightened into a D-dimensional vector. We will treat this vector as the representative of our image in the aforementioned joint semantic space with the sentences. The vector is then asked to reconstruct the original image. The reconstruction is just a fully connected layer, which maps the D-dimensional vector into the space whose dimension is equal to the number of input neurons into the network, and then applies an activation function. The resulting output is compared to the original image and their L_2 distance is used as the training objective of the whole neural network. In this fully connected layer,

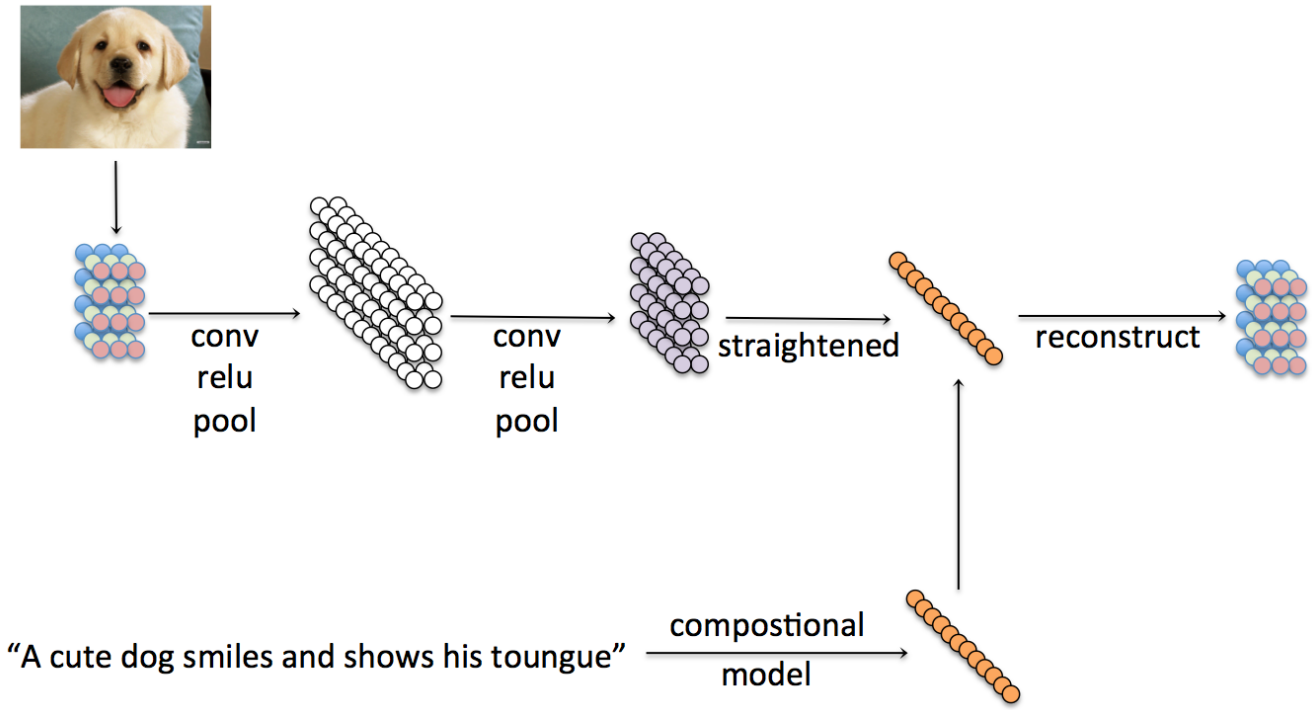

Figure 1. Autoencoder architecture to learn image embeddings.

we use the logistic sigmoid activation function

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{1}$$

instead of the rectified linear unit, for two reasons. First, as a convention, we normalize the pixel values of the input image to values in $[0, 1]$, which is also the range of $\sigma(\cdot)$. Compared to the rectified linear unit, which allows in the activated output to be arbitrary large, and hyperbolic tangent, which has range $[-1, 1]$, $\sigma(\cdot)$ is more appropriate since it already puts the reconstructed output of the network in the same range with the autoencoder target. Second, one of the most serious problem with sigmoid is that backpropagation using it through many layers shrinks the gradient due to the term $\sigma \cdot (1 - \sigma)$, but here we only have one fully connected layer, so this would not be a problem.

Our training images come with some human-annotated captions. The natural assumption is that these captions and the image describe the same subjects, so their embeddings should be close to each other in the joint semantic space. To encourage this phenomenon, we augment our training objective with the L_2 distance between the second pool layer's output (the D-dimensional vector) with the embedding of a sentence randomly sampled from the given sent of captions. Overall, for a sample S from the training image I and it sample caption $C(S)$, our model learns an embedding $E(S)$, based on which it produces the reconstructed version $R(S)$, and our objective function for S is

$$J(S) = \frac{1}{2}\left(\|S - R(S)\|_2^2 + \lambda \|E(S) - C(S)\|_2^2\right), \tag{2}$$

where $\|\cdot\|_2$ is the L_2 distance and $\lambda > 0$ is the parameter that controls the focus of the model. When λ is large, the model emphasizes that the image embedding is the same as its caption's; when λ is small, the model emphasizes that the reconstructed image is faithful to the input sample.

At test time, there are two separated procedures. If our text model is given a caption, which is essentially an arbitrary sequence of words, it returns the embedding of the sequence depending on the choice of compositional model, either ABOW or PV. As discussed, this embedding will be a D-dimensional vector. If our vision model is given an image, it first samples a couple of contiguous regions from the image. The sample size specified at train time will be followed. Each of the sample is then passed as input to the learned convolutional neural network to compute the distributed representation of the corresponding sample, i.e. the output of the second pool layer. Finally, these embeddings are averaged to obtain the embedding of the image.

4. Experiments

In this section, we describe the experiments to evaluate our model's performance.

4.1. Dataset

English corpus. Our textual training set is the monolingual English corpus from Europarl v7 by [14]. It consists of $2.2M$ sentences. We preprocess the data by lowercasing all characters in the corpus and filtering out all words that appear less than 5 times. The resulting vocabulary consists of $46K$ words.

Image dataset. Our image dataset is the *Flickr8k* provided by [11]. It consists of $8K$ images of various sizes, each comes with five captions collected from Amazon Mechanical Turk. The dataset is also separated into $6K$ train images, $1K$ dev images and $1K$ test images. We did minimal preprocessing to the captions by changing all the lectures to lowercase. The vocabulary resulting from the preprocesed captions turns out to be a subset of the vocabulary we learned from Europarl, with the exception of some proper nouns such as names of people and places, which we address to with a special token $\langle unk \rangle$, whose embedding is the zero vector.

4.2. Training details and procedures

Since *Flickr8k* images have various sizes, we have to sample their subregions to train on. We fixed our sample size to be 64×64. This size is relatively small compared to the original size of many images in our data (typically 200-300) and hence raises the concern that the sampling process might not cover the original image. We simply overcome the setback by sampling through more epochs. The tradeoff is that supposed we take samples of size $W \times H$ from the image. Then, after two convolutional-rectified linear-pool combined layers, we end up with an image embedding of size $D = c \times W/4 \times H/4$, where c is the number of receptive fields in the second convolutional layer of our network. We want this resulting vector to live in the same D-dimensional space with the word (and sentence) embeddings. Reasonable choices for word and sentence embeddings size are 50-1000, and the choice $W = H = 64$ along with $c = 4$ gives $D = 1024$. Note that we also do not want smaller c because convolutional with too few receptive fields are less favorable, as pointed out by [3]. Last but not least, our first convolutional layer has 64 receptive fields.

We implement both of our compostional model and image model in C++. We speedup our training process with multithreading code. Our model is trained using asynchronous gradient descent [33] for 50 epochs. We have 10 samples for each image. In each epoch, for each image, the vision model randomly picks a sample and a caption and updates its corresponding parameters to minimize the training objective with respect to the image. After each epoch, we anneal our learning rate to encourage faster convergence. We regularize our training procedure by dropout [29] with probability p at each convolutional layer. Using the training objective function evaluated on the dev images provided by *Flickr8k*, we tune our three hyper-parameters: the relative strength λ of reconstruction error with respect to caption error, the learning rate α and the dropout probability p. Our final asset is $\lambda = 2.8$, $\alpha = 10^{-5}$ and $p = 0.75$.

4.3. Evaluation and result

We evaluate our learned model on two tasks: image caption ranking and image search. In the prior task, the learned model is given one image and several captions and is asked to rank the captions according to their degrees of relevance to it. In the latter task, the converse process is defined: the model is given a caption query and has to rank the images in its database also according to their degrees of relevance to the query. To obtain the images and queries, we use the *Flickr8k*'s $1K$ test images along with their $5k$ captions as our image and caption database respectively. Naturally, our metric for relevance is the L_2 distance between the distributed representations the image and the caption, smaller means more relevant. Our metric of evaluation is Recall@K, the number of correct images or documents for the query in the most relevant K items. This is the same procedure as in previous work such as [6, 12, 28]. To evaluate our model, we also compare its result against those of its handicapped versions, in particular the vision model that is not trained along with the errors from the textual model. In Table 1, we summarize our result and compare it against the baselines.

5. Discussion

5.1. Learning multimodal knowledge

Compared to state of the art results on the same tasks of ranking images and their captions, the performance of our models is considered modest. However, our results have some interesting implications. First and perhaps most importantly, we have shown that it is possible to integrate the autoencoder objective with a guiding objective that directs the image embeddings learned by our model to desired positions in the semantic space. Indeed, the results in Table for has shown that the handicapped version of our model, without the error signals from textual data, performs no better than random ranking. This agrees with our intuition, because even though the model might have learned some meaningful features from the autoencoder, there is no hope that its interpretation on image semantics is similar to that of a textual model trained independently.

On the other hand, compared to modern autoencoder networks such as [17, 22], our vision model with two convolutional-rectified linear-pool is very small, implying the weakness in learning abstract high-level features. However, the tradeoff is that we have a decent textual model: the *word2vec* word embeddings and the PV model are both state of the art methods in the field [23, 16], and in our ex-

Model	Annotation Ranking			Image Ranking		
	R@1	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**
Random ranking	0.1	0.6	1.1	0.1	0.5	1.0
Socher et al. [28]	4.5	18.0	28.6	6.1	18.5	29.0
DiViSE [6]	4.8	16.5	27.3	5.9	20.1	29.6
SDT-RNN [28]	6.0	22.7	34.0	6.6	21.6	31.7
Karpathy et al. [12]	**12.6**	**32.9**	**44.0**	**9.7**	**29.6**	**42.5**
Autoencoder †	0.2	0.5	1.3	0.1	0.6	1.4
† + ABOW	2.0	9.7	12.1	2.5	10.7	13.8
† + PV	3.1	11.3	15.2	4.2	12.2	17.6

Table 1. Recall@K results on *Flickr8k*. Autoencoder is our visual model without error signals from the L_2 distance between second pool layer and the caption, and the compositional model is ABOW.

λ	Annotation R@1	Image R@1
0.5	0.7	1.8
1.0	1.2	2.2
2.0	2.4	3.7
3.5	**3.1**	**4.1**
10	2.6	3.2

Table 2. Performance of Autoencoder+PV models with different λ. Too large values of λ is not as good because with those values, the model just simply learns to map images into the embeddings of their captions. This is nice, but in our settings, at each training epoch, the model only sees one sample and one caption per image, so they might be inconsistent, particularly with small sample size.

periments, these models are trained on carefully processed corpus. Thus, the textual knowledge of our model is decent. This knowledge is used to guide the training of the vision model, giving $15\times$ improve of results from the autoencoder alone to autoencoder with guiding PV model, as shown. We quantify this claim with the comparisons of the results across the Autoencoder+PV model, with different values of λ, the parameters that control whether our objective should focus on optimizing the autoencoder error or the embedding discrepancy error.

5.2. Pros and cons of image sampling

As discussed earlier, we sample our input images to so that it can be input into a network with hard-coded input size. We have identified two main the disadvantages of our sampling method: (1) the samples might not cover the whole image and more seriously, leaving out important information; (2) the network cannot operate on images whose either width our height is less than the sample size (ours is 64). On the other hand, the advantage is that we can manually retrieve our desired parts of the image and investigate the the knowledge learned by our model. For example, we plot the distributed representations of the image samples shown in Figure 2 after PCA them into 2-dimensional plane

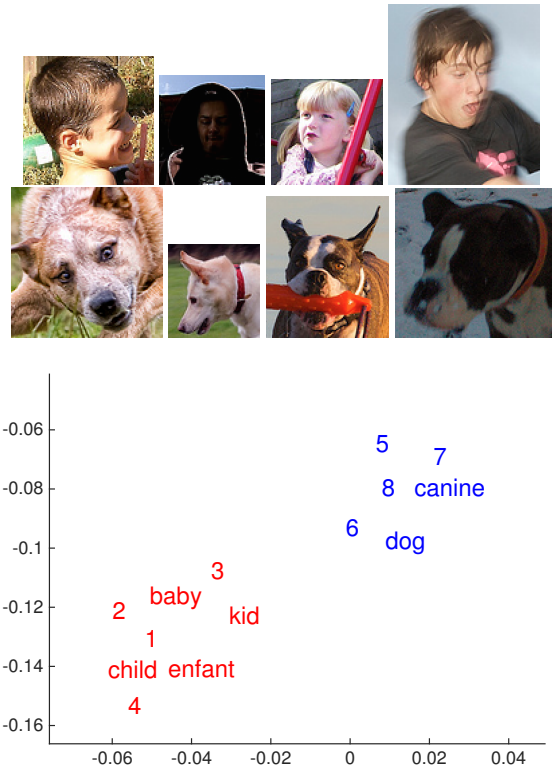

Figure 2. First row: crops of children faces, numbered 1-4. Second row: crops of dog faces, numbered 5-8. We plot the distributed representations of these images and those of the words: *baby*, *kid*, *child*, *dog*, *enfant*, and *canine*. The last two words are not presented in any caption of *Flickr8k*, but they are in our English corpus, so our model still learns their word embeddings. The plot is self-explained.

using t-SNE [31].

5.3. Directions for improvement

Finally, our framework, autoencoder with textual error signals, can give better performance, should the architec-

ture is more sophisticated. Two improvements that be made include: (1) more convolutional-rectified linear-pool combined layers and (2) more fully connected layers in the reconstruction process. The first direction also allows the output of the last pool layer to have smaller size relative to the input layer, leading to the feasibility of larger sample size and perhaps, more receptive field in the last convolutional layer, which leads to better capability of the model to abstract its visual knowledge. However, given the time constraint and the scope of this project, we leave these directions as future work.

6. Conclusion

In summary, we have described our approach to learn the distributed representations of images and arbitrary word sequences. We have also shown that the representations of images of similar objects and of text sequences of the same meanings are related by their small L_2 distance from each other in the joint semantic space. Our work implies that multimodal knowledge from decent textual models can indeed aid the learning process of a weak visual model. Moving forward, we have proposed numerous ideas to improve the current model's performance. We believe that if our textual model is combined with a strong visual model, it is possible to achieve much better performances on various tasks.

Acknowledgement

We thank CS231N staffs, especially Professor Fei-Fei Li and Andreij Karpathy for offering the class. This work would never have existed without their instruction for us on a cutting-edge techniques. For us, it has been a challenging and stressful yet enjoyable and beneficial quarter. We hope that similar and more advanced classes in the field will be offered.

References

[1] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *JMLR*, 3:1137–1155, 2003.

[2] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, U. D. Montral, and M. Qubec. Greedy layer-wise training of deep networks. In *In NIPS*. MIT Press, 2007.

[3] A. Coates and A. Y. Ng. Selecting receptive fields in deep networks. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2528–2536. Curran Associates, Inc., 2011.

[4] R. Collobert. Deep learning for efficient discriminative parsing. In *AISTATS*, 2011.

[5] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537, 2011.

[6] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2121–2129. Curran Associates, Inc., 2013.

[7] S. Gouws, Y. Bengio, and G. Corrado. Bilbowa: Fast bilingual distributed representations without word alignments. *arXiv preprint arXiv:1410.2455*, 2014.

[8] K. M. Hermann and P. Blunsom. Multilingual Models for Compositional Distributional Semantics. In *ACL*, 2014.

[9] G. E. Hinton. Rectified linear units improve restricted boltzmann machines vinod nair.

[10] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[11] M. Hodosh, P. Young, and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *J. Artif. Intell. Res. (JAIR)*, 47:853–899, 2013.

[12] A. Karpathy, A. Joulin, and F. F. F. Li. Deep fragment embeddings for bidirectional image sentence mapping. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1889–1897. Curran Associates, Inc., 2014.

[13] A. Klementiev, I. Titov, and B. Bhattarai. Inducing crosslingual distributed representations of words. In *COLING*, 2012.

[14] P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *MT Summit*, 2005.

[15] H. Larochelle and Y. Bengio. Classification using discriminative restricted boltzmann machines. In *In ICML 08: Proceedings of the 25th international conference on Machine learning. ACM*, 2008.

[16] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In T. Jebara and E. P. Xing, editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196. JMLR Workshop and Conference Proceedings, 2014.

[17] Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng. Building high-level features using large scale unsupervised learning. In

J. Langford and J. Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 81–88, New York, NY, USA, July 2012. Omnipress.

[18] Q. V. Le, J. Ngiam, Z. Chen, D. Chia, P. W. Koh, and A. Y. Ng. Tiled convolutional neural networks. In *In NIPS, in press*, 2010.

[19] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[20] M. Lin, Q. Chen, and S. Yan. Network in network. *CoRR*, abs/1312.4400, 2013.

[21] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[22] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *Proceedings of the 21th International Conference on Artificial Neural Networks - Volume Part I*, ICANN'11, pages 52–59, Berlin, Heidelberg, 2011. Springer-Verlag.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *ICLR*, 2013.

[24] T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In *NAACL-HLT*, 2013.

[25] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In L. Getoor and T. Scheffer, editors, *ICML*, pages 689–696. Omnipress, 2011.

[26] R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *NIPS*, 2011.

[27] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *EMNLP*, 2012.

[28] R. Socher, A. Karpathy, V. Q. Le, D. C. Manning, and Y. A. Ng. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association of Computational Linguistics – Volume 2, Issue 1*, pages 207–218, 2014.

[29] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[30] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. In *ACL*, 2010.

[31] L. van der Maaten and G. E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[32] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: Learning to rank with joint word-image embeddings. *Mach. Learn.*, 81(1):21–35, Oct. 2010.

[33] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola. Parallelized stochastic gradient descent. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2595–2603. Curran Associates, Inc., 2010.