A Multiplicative Model for Learning Distributed Text-Based Attribute Representations

Authored by:

Ruslan R. Salakhutdinov
Ryan Kiros
Richard Zemel

Abstract

In this paper we propose a general framework for learning distributed representations of attributes: characteristics of text whose representations can be jointly learned with word embeddings. Attributes can correspond to a wide variety of concepts, such as document indicators (to learn sentence vectors), language indicators (to learn distributed language representations), meta-data and side information (such as the age, gender and industry of a blogger) or representations of authors. We describe a third-order model where word context and attribute vectors interact multiplicatively to predict the next word in a sequence. This leads to the notion of conditional word similarity: how meanings of words change when conditioned on different attributes. We perform several experimental tasks including sentiment classification, cross-lingual document classification, and blog authorship attribution. We also qualitatively evaluate conditional word neighbours and attribute-conditioned text generation.

1 Paper Body

Distributed word representations have enjoyed success in several NLP tasks [1, 2]. More recently, the use of distributed representations have been extended to model concepts beyond the word level, such as sentences, phrases and paragraphs [3, 4, 5, 6], entities and relationships [7, 8] and embeddings of semantic categories [9, 10]. In this paper we propose a general framework for learning distributed representations of attributes: characteristics of text whose representations can be jointly learned with word embeddings. The use of the word attribute in this context is general. Table 1 illustrates several of the experiments we perform along with the corresponding notion of attribute. For example, an attribute can represent an indicator of the current sentence or language being processed. This allows us to learn sentence and language vectors, similar to the proposed model of [6]. Attributes can also correspond to side information, or metadata associated with text. For instance, a collection of blogs may come

with information about the age, gender or industry of the author. This allows us to learn vectors that can capture similarities across metadata based on the associated body of text. The goal of this work is to show that our notion of attribute vectors can achieve strong performance on a wide variety of NLP related tasks. In particular, we demonstrate strong quantitative performance on three highly diverse tasks: sentiment classification, cross-lingual document classification, and blog authorship attribution. To capture these kinds of interactions between attributes and text, we propose to use a third-order model where attribute vectors act as gating units to a word embedding tensor. That is, words are represented as a tensor consisting of several prototype vectors. Given an attribute vector, a word embedding matrix can be computed as a linear combination of word prototypes weighted by the attribute representation. During training, attribute vectors reside in a separate lookup table which can be jointly learned along with word features and the model parameters. This type of three-way 1

Table 1: Summary of tasks and attribute types used in our experiments. The first three are quantitative while the second three are qualitative. Task Sentiment Classification Cross-Lingual Classification Authorship Attribution Conditional Text Generation Structured Text Generation Conditional Word Similarity

Dataset Sentiment Treebank RCV1/RCV2 Blog Corpus Gutenberg Corpus Gutenberg Corpus Blogs & Europarl

Attribute type Sentence Vector Language Vector Author Metadata Book Vector Part of Speech Tags Author Metadata / Language

interaction can be embedded into a neural language model, where the three-way interaction consists of the previous context, the attribute and the score (or distribution) of the next word after the context. Using a word embedding tensor gives rise to the notion of conditional word similarity. More specifically, the neighbours of word embeddings can change depending on which attribute is being conditioned on. For example, the word ?joy? when conditioned on an author with the industry attribute ?religion? appears near ?rapture? and ?god? but near ?delight? and ?comfort? when conditioned on an author with the industry attribute ?science?. Another way of thinking of our model would be the language analogue of [11]. They used a factored conditional restricted Boltzmann machine for modelling motion style defined by real or continuous valued style variables. When our factorization is embedded into a neural language model, it allows us to generate text conditioned on different attributes in the same manner as [11] could generate motions from different styles. As we show in our experiments, if attributes are represented by different books, samples generated from the model learn to capture associated writing styles from the author. Furthermore, we demonstrate a strong performance gain for authorship attribution when conditional word representations are used. Multiplicative interactions have also been previously incorporated into neural language models. [12] introduced a multiplicative model where images are used for gating word representations. Our framework can be seen as a generalization of [12] and in the context of their work an attribute would correspond to a fixed representation of an image. [13] introduced a multiplicative recurrent neural network

for generating text at the character level. In their model, the character at the current timestep is used to gate the network?s recurrent matrix. This led to a substantial improvement in the ability to generate text at the character level as opposed to a non-multiplicative recurrent network.

2

Methods

In this section we describe the proposed models. We first review the log-bilinear neural language model of [14] as it forms the basis for much of our work. Next, we describe a word embedding tensor and show how it can be factored and introduced into a multiplicative neural language model. This is concluded by detailing how our attribute vectors are learned. 2.1

Log-bilinear neural language models

The log-bilinear language model (LBL) [14] is a deterministic model that may be viewed as a feedforward neural network with a single linear hidden layer. Each word w in the vocabulary is represented as a K-dimensional real-valued vector rw ? RK . Let R denote the V ? K matrix of word representation vectors where V is the vocabulary size. Let (w1 , . . . wn?1) be a tuple of n ? 1 words where n ? 1 is the context size. The LBL model makes a linear prediction of the next word representation as n?1 X ? r= C(i) rwi , (1) i=1 (i)

where C , i = 1, . . . , n ? 1 are K ? K context parameter matrices. Thus, ? r is the predicted representation of rwn . The conditional probability P (wn = i—w1:n?1) of wn given w1 , . . . , wn?1 is exp(? rT ri + bi) P (wn = i—w1:n?1) = PV , rT rj + bj) j=1 exp(? where b ? RV is a bias vector. Learning can be done using backpropagation. 2

(2)

(a) NLM

(b) Multiplicative NLM

(c) Multiplicative NLM with language switch

Figure 1: Three different formulations for predicting the next word in a neural language model. Left: A standard neural language model (NLM). Middle: The context and attribute vectors interact via a multiplicative interaction. Right: When words are unshared across attributes, a one-hot attribute vector gates the factors-to-vocabulary matrix. 2.2

A word embedding tensor

Traditionally, word representation matrices are represented as a matrix R ? RV ?K , such as in the case of the log-bilinear model. Throughout this work, we instead represent words as a tensor T ? RV ?K?D where D corresponds to the number of tensor slices. Given an attribute vector PD x ? RD , we can compute attribute-gated word representations as T x = i=1 xi T (i) i.e. word representations with respect to x are computed as a linear combination of slices weighted by each component xi of x. It is often unnecessary to use a fully unfactored tensor. Following [15, 16], we re-represent T in terms of three matrices Wf k ? RF ?K , Wf d ? RF ?D and Wf v ? RF ?V , such that T x = (Wf v)¿ ? diag(Wf d x) ? Wf k , (3) where diag(?) denotes the matrix with its argument on the diagonal. These matrices are parametrized by a pre-chosen number of factors F . 2.3

Multiplicative neural language models

We now show how to embed our word representation tensor T into the log-bilinear neural language model. Let E = (Wf k)¿ Wf v denote a ?folded? K ? V matrix of word embeddings. Given the context w1 , . . . , wn?1 , the predicted next word representation ? r is given by ? r=

n?1 X

C(i) E(:, wi),

(4)

i=1

where E(:, wi) denotes the column of E for the word representation of wi and C(i) , i = 1, . . . , n?1 are K ? K context matrices. Given a predicted next word representation ? r, the factor outputs are f = (Wf k? r) ? (Wf d x), (5) where ? is a component-wise product. The conditional probability P (wn = i—w1:n?1 , x) of wn given w1 , . . . , wn?1 and x can be written as

exp (Wf v (:, i))¿ f + bi P (wn = i—w1:n?1 , x) = PV . fv ¿ j=1 exp (W (:, j)) f + bj Here, Wf v (:, i) denotes the column of Wf v corresponding to word i. In contrast to the log-bilinear model, the matrix of word representations R from before is replaced with the factored tensor T , as shown in Fig. 1. 2.4

Unshared vocabularies across attributes

Our formulation for T assumes that word representations are shared across all attributes. In some cases, words may only be specific to certain attributes and not others. An example of this is crosslingual modelling, where it is necessary to have language specific vocabularies. As a running example, consider the case where each attribute corresponds to a language representation vector. Let 3

Table 2: Samples generated from the model when conditioning on various attributes. For the last example, we condition on the average of the two vectors (symbol ¡#¿ corresponds to a number). Attribute Bible Caesar 1 2

(Bible + Caesar)

Sample ¡#¿ : ¡#¿ for thus i enquired unto thee , saying , the lord had not come unto him . ¡#¿ : ¡#¿ when i see them shall see me greater am that under the name of the king on israel . to tell vs pindarus : shortly pray , now hence , a word . comes hither , and let vs exclaim once by him fear till loved against caesar . till you are now which have kept what proper deed there is an ant ? for caesar not wise cassi let our spring tiger as with less ; for tucking great fellowes at ghosts of broth . industrious time with golden glory employments . ¡#¿ : ¡#¿ but are far in men soft from bones , assur too , set and blood of smelling , and there they cost , i learned : love no guile his word downe the mystery of possession

x denote the attribute vector for language ‘ and x0 for language ‘0 (e.g. English and French). We can then compute language-specific word representations T ‘ by breaking up our decomposition into language dependent and independent components (see Fig. 1c): T ‘ = (W‘f v)¿ ? diag(Wf d x) ? Wf k ,

(6)

where (W‘f v)¿ is a V‘ ? F language specific matrix. The matrices Wf d and Wf k do not depend on the language or the vocabulary, whereas (W‘f v)¿ is language specific. Moreover, since each language may have a different sized

vocabulary, we use V‘ to denote the vocabulary size of language ‘. Observe that this model has an interesting property in that it allows us to share statistical strength across word representations of different languages. In particular, we show in our experiments how we can improve cross-lingual classification performance between English and German when a large amount of parallel data exists between English and French and only a small amount of parallel data exists between English and German. 2.5

Learning attribute representations

We now discuss how to learn representation vectors x. Recall that when training neural language models, the word representations of w1 , ... , wn?1 are updated by backpropagating through the word embedding matrix. We can think of this as being a linear layer, where the input to this layer is a one-hot vector with the i-th position active for word wi . Then multiplying this vector by the embedding matrix results in the word vector for wi . Thus the columns of the word representations matrix consisting of words from w1 , ... , wn?1 will have non-zero gradients with respect to the loss. This allows us to consistently modify the word representations throughout training. We construct attribute representations in a similar way. Suppose that L is an attribute lookup table, where x = f (L(:, x)) and f is an optional non-linearity. We often use a rectifier non-linearity in order to keep x sparse and positive, which we found made training much more stable. Initially, the entries of L are generated randomly. During training, we treat L in the same way as the word embedding matrix. This way of learning language representations allows us to measure how ?similar? attributes are as opposed to using a one-hot encoding of attributes for which no such similarity could be computed. In some cases, attributes that are available during training may not also be available at test time. An example of this is when attributes are used as sentence indicators for learning representations of sentences. To accommodate for this, we use an inference step similar to that proposed by [6]. That is, at test time all the network parameters are fixed and stochastic gradient descent is used for inferring the representation of an unseen attribute vector.

3

Experiments

In this section we describe our experimental evaluation and results. Throughout this section we refer to our model as Attribute Tensor Decomposition (ATD). All models are trained using stochastic gradient descent with an exponential learning rate decay and linear (per epoch) increase in momentum. We first demonstrate initial qualitative results to get a sense of the tasks our model can perform. For these, we use the small project Gutenberg corpus which consists of 18 books, some of which have the same author. We first trained a multiplicative neural language model with a context size of 5, 4

Table 3: A modified version of the game Mad Libs. Given an initialization, the model is to generate the next 5 words according to the part-of-speech sequence (note that these are not hard constraints). [DT, NN, IN, DT, JJ] the meaning of life is... the cure of the bad the truth of the good a penny for the fourth the globe of those modern all man upon the same

[TO, VB, VBD, JJS, NNS] my greatest accomplishment is... to keep sold most wishes to make manned most magnificent to keep wounded best nations to be allowed best arguments to be mentioned most people

[PRP, NN, ?,? , JJ, NN] i could not live without... his regard , willing tenderness her french , serious friend her father , good voice her heart , likely beauty her sister , such character

Table 4: Classification accuracies on various tasks. Left: Sentiment classification on the treebank dataset. Competing methods include the Neural Bag of words (NBoW) [5], Recursive Network (RNN) [17], Matrix-Vector Recursive Network (MV-RNN) [18], Recursive Tensor Network (RTNN) [3], Dynamic Convolutional Network (DCNN) [5] and Paragraph Vector (PV) [6]. Right: Cross-lingual classification on RCV2. Methods include statistical machine translation (SMT), IMatrix [19], Bag-of-words autoencoders (BAE-*) [20] and BiCVM, BiCVM+ [21]. The use of ?+? on cross-lingual tasks indicate the use of a third language (French) for learning embeddings. Method SVM BiNB NBoW RNN MVRNN RTNN DCNN PV ATD

Fine-grained 40.7% 41.9% 42.4% 43.2% 44.4% 45.7% 48.5% 48.7% 45.9%

Positive / Negative 79.4% 83.1% 80.5% 82.4% 82.9% 85.4% 86.8% 87.8% 83.3%

Method SMT I-Matrix BAE-cr BAE-tree BiCVM BiCVM+ BAE-corr ATD ATD+

EN ? DE 68.1% 77.6% 78.2% 80.2% 83.7% 86.2% 91.8% 80.8% 83.4%

DE ? EN 67.4% 71.1% 63.6% 68.2% 71.4% 76.9% 72.8% 71.8% 72.9%

where each attribute is represented as a book. This results in 18 learned attribute vectors, one for each book. After training, we can condition on a book vector and generate samples from the model. Table 2 illustrates some the generated samples. Our model learns to capture the ?style? associated with different books. Furthermore, by conditioning on the average of book representations, the model can generate reasonable samples that represent a hybrid of both attributes, even though such attribute combinations were not observed during training. Next, we computed POS sequences from sentences that occur in the training corpus. We trained a multiplicative neural language model with a context size of 5 to predict the next word from its context, given knowledge of the POS tag for the next word. That is, we model P (wn = i—w1:n?1 , x) where x denotes the POS tag for word wn . After training, we gave the model an initial input and a POS sequence and proceeded to generate samples. Table 3 shows some results for this task. Interestingly, the model can generate rather funny and poetic completions to the initial context. 3.1

Sentiment classification

Our first quantitative experiments are performed on the sentiment treebank of [3]. A common challenge for sentiment classification tasks is that the global sentiment of a sentence need not correspond to local sentiments exhibited in subphrases of the sentence. To address this issue, [3] collected annotations from the movie reviews corpus of [22] of all subphrases extracted from a sentence parser. By incorporating local sentiment into their recursive architectures, [3] was able to obtain significant performance gains with recursive networks over

bag of words baselines. We follow the same experimental procedure proposed by [3] for which evaluation is reported on two tasks: fine-grained classification of categories {very negative, negative, neutral, positive, very positive } and binary classification {positive, negative }. We extracted all subphrases of sentences that occur in the training set and used these to train a multiplicative neural language model. Here, each attribute is represented as a sentence vector, as in [6]. In order to compute subphrases for unseen sentences, we apply an inference procedure similar to [6], where the weights of the network are frozen and gradient descent is used to infer representations for each unseen vector. We trained a logistic regression classifier using all training subphrases in the training set. At test time, we infer a representation for a new sentence which is used for making a review prediction. We used a context 5

size of 8, 100 dimensional word vectors initialized from [2] and 100 dimensional sentence vectors initialized by averaging vectors of words from the corresponding sentence. Table 4, left panel, illustrates our results on this task in comparison to all other proposed approaches. Our results are on par with the highest performing recursive network on the fine-grained task and outperforms all bag-of-words baselines and recursive networks with the exception of the RTNN on the binary task. Our method is outperformed by the two recently proposed approaches of [5] (a convolutional network trained on sentences) and Paragraph Vector [6]. 3.2

Cross-lingual document classification

We follow the experimental procedure of [19], for which several existing baselines are available to compare our results. The experiment proceeds as follows. We first use the Europarl corpus [23] for inducing word representations across languages. Let S be a sentence with words w in language ‘ and let x be the corresponding language vector. Let X X v‘ (S) = T ‘ (:, w) = (W‘f v (:, w))¿ ? diag(Wf d x) ? Wf k (7) w?S

w?S

denote the sentence representation of S, defined as the sum of language conditioned word representations for each w ? S. Equivalently we define a sentence representation for the translation S 0 of S denoted as v‘0 (S 0). We then optimize the following ranking objective:

XX

2

2

2 minimize max 0, ? + v‘ (S) ? v‘0 (S 0) 2 ? v‘ (S) ? v‘0 (Ck) 2 + ? ? 2 ?

S

k

subject to the constraints that each sentence vector has unit norm. Each Ck is a constrastive (nontranslation) sentence of S and ? denotes all model parameters. This type of cross-language ranking loss was first used by [21] but without the norm constraint which we found significantly improved the stability of training. The Europarl corpus contains roughly 2 million parallel sentence pairs between English and German as well as English and French, for which we induce 40 dimensional word representations. Evaluation is then performed on

English and German sections of the Reuters RCV1/RCV2 corpora. Note that these documents are not parallel. The Reuters dataset contains multiple labels for each document. Following [19], we only consider documents which have been assigned to one of the top 4 categories in the label hierarchy. These are CCAT (Corporate/Industrial), ECAT (Economics), GCAT (Government/Social) and MCAT (Markets). There are a total of 34,000 English documents and 42,753 German documents with vocabulary sizes of 43614 English words and 50,110 German words. We consider both training on English and evaluating on German and vice versa. To represent a document, we sum over the word representations of words in that document followed by a unit-ball projection. Following [19] we use an averaged perceptron classifier. Classification accuracy is then evaluated on a held-out test set in the other language. We used a monolingual validation set for tuning the margin ?, which was set to ? = 1. Five contrastive terms were used per example which were randomly assigned per epoch. Table 4, right panel, shows our results compared to all proposed methods thus far. We are competitive with the current state-of-the-art approaches, being outperformed only by BiCVM+ [21] and BAE-corr [20] on EN ? DE. The BAE-corr method combines both a reconstruction term and a correlation regularizer to match sentences, while our method does not consider reconstruction. We also performed experimentation on a low resource task, where we assume the same conditions as above with the exception that we only use 10,000 parallel sentence pairs between English and German while still incorporating all English and French parallel sentences. For this task, we compare against a separation baseline, which is the same as our model but with no parameter sharing across languages (and thus resembles [21]). Here we achieve 74.7% and 69.7% accuracies (EN?DE and DE?EN) while the separation baseline obtains 63.8% and 67.1%. This indicates that parameter sharing across languages can be useful when only a small amount of parallel data is available. Figure 2 further shows t-SNE embeddings of English-German word pairs.1 Another interesting consideration is whether or not the learned language vectors can capture any interesting properties of various languages. To look into this, we trained a multiplicative neural language model simultaneously on 5 languages: English, French, German, Czech and Slovak. To our knowledge, this is the most languages word representations have been jointly learned on. We 1

We note that Germany and Deutschland are nearest neighbours in the original space.

6

(a) Months

(b) Countries

5 4 3 2 1

0.3 0.2 0.1 0.0 ? 0.1 uncondit ioned ATD

? 0.2 0

(a) Correlation matrix

uncondit ioned ATD LBL condit ioned ATD

6

Inferred at t ribut es difference

Im provem ent over init ial m odel

Figure 2: t-SNE embeddings of English-German word pairs learned from Europarl.

5

10 25 50 100 # Docum ent s (t housands)

382

5

10 25 50 100 # Docum ent s (t housands)

382

(b) Effect of conditional embeddings (c) Effect of inferring attribute vectors

Figure 3: Results on the Blog classification corpus. For the middle and right plots, each pair of same coloured bars corresponds to the non-inclusion or inclusion of inferred attribute vectors, respectively. computed a correlation matrix from the language vectors, illustrated in Fig. 3a. Interestingly, we observe high correlation between Czech and Slovak representations, indicating that the model may have learned some notion of lexical similarity. That being said, additional experimentation for future work is necessary to better understand the similarities exhibited through language vectors. 3.3

Blog authorship attribution

For our final task, we use the Blog corpus of [24] which contains 681,288 blog posts from 19,320 authors. For our experiments, we break the corpus into two separate datasets: one containing the 1000 most prolific authors (most blog posts) and the other containing all the rest. Each author comes with an attribute tag corresponding to a tuple (age, gender, industry) indicating the age range of the author (10s, 20s or 30s), whether the author is male or female, and what industry the author works in. Note that industry does not necessary correspond to the topic of blog posts. We use the dataset of non-prolific authors to train a multiplicative language model conditioned on an attribute tuple of which there are 234 unique tuples in total. We used 100 dimensional word vectors initialized from [2], 100 dimensional attribute vectors with random initialization and a context size of 5. A 1000-way classification task is then performed on the prolific author subset and evaluation is done using 10-fold cross-validation. Our initial experimentation with baselines found that tf-idf performs well on this dataset (45.9% accuracy). Thus, we consider how much we can improve on the tf-idf baseline by augmenting word and attribute features. For the first experiment, we determine the effect conditional word embeddings have on classification performance, assuming attributes are available at test time. For this, we compute two embedding matrices from a trained ATD model, one without and with attribute knowledge: unconditioned ATD : conditioned ATD :

(Wf v)¿ Wf k

(8)

(Wf v)¿ ? diag(Wf d x) ? Wf k .

(9)

We represent a blog post as the sum of word vectors projected to unit norm and augment these with tf-idf features. As an additional baseline we include a log-bilinear language model [14]. 2 Figure 3b illustrates the results from which

we observe that conditioned word embeddings are significantly more discriminative over word embeddings computed without knowledge of attribute vectors. 2

The log-bilinear model has no concept of attributes.

7

Table 5: Results from a conditional word similarity task using Blog attributes and language vectors. Query,A,B school f/10/student m/20/tech journal f/10/student m/30/adv. create f/30/arts f/30/internet joy m/30/religion m/20/science cool m/10/student f/10/student

Common work church college diary blog webpage build develop maintain happiness sadness pain nice funny awesome

Unique to A choir prom skool project book yearbook provide acquire generate rapture god heartbreak beautiful amazing neat

Unique to B therapy tech job zine app referral compile follow analyse delight comfort soul sexy hott lame

English january june october market markets internal war weapons global said stated told two two-thirds both

French janvier decembre juin marche marches interne guerre terrorisme mondaile dit disait declare deux deuxieme seconde

German januar dezember juni markt binnenmarktes marktes krieg globale krieges sagte gesagt sagten zwei beiden zweier

For the second experiment, we determine the effect of inferring attribute vectors at test time if they are not assumed to be available. To do this, we train a logistic regression classifier within each fold for predicting attributes. We compute an inferred vector by averaging each of the attribute vectors weighted by the log-probabilities of the classifier. In Fig. 3c we plot the difference in performance when an inferred vector is augmented vs. when it is not. These results show consistent, albeit small improvement gains when attribute vectors are inferred at test time. To get a better sense of the attribute features learned from the model, the supplementary material contains a t-SNE embedding of the learned attribute vectors. Interestingly, the model learns features which largely isolate the vectors of all teenage bloggers independent of gender and topic. 3.4

Conditional word similarity

One of the key properties of our tensor formulation is the notion of conditional word similarity, namely how neighbours of word representations change depending on the attributes that are conditioned on. In order to explore the effects of this, we performed two qualitative comparisons: one using blog attribute vectors and the other with language vectors. These results are illustrated in Table 5. For the first comparison on the left, we chose two attributes from the blog corpus and a query word. We identify each of these attribute pairs as A and B. Next, we computed a ranked list of the nearest neighbours (by cosine similarity) of words conditioned on each attribute and identified the top 15 words in each. Out of these 15 words, we display the top 3 words which are common to both ranked lists, as well as 3 words that are unique to a specific attribute. Our results illustrate that the model can capture distinctive notions of word similarities depending on which attributes are being conditioned. On

the right of Table 5, we chose a query word in English (italicized) and computed the nearest neighbours when conditioned on each language vector. This results in neighbours that are either direct translations of the query word or words that are semantically similar. The supplementary material includes additional examples with nearest neighbours of collocations.

4

Conclusion

There are several future directions from which this work can be extended. One application area of interest is in learning representations of authors from papers they choose to review as a way of improving automating reviewer-paper matching [25]. Since authors contribute to different research topics, it might be more useful to instead consider a mixture of attribute vectors that can allow for distinctive representations of the same author across research areas. Another interesting application is learning representations of graphs. Recently, [26] proposed an approach for learning embeddings of nodes in social networks. Introducing network indicator vectors could allow us to potentially learn representations of full graphs. Finally, it would be interesting to train a multiplicative neural language model simultaneously across dozens of languages. Acknowledgments We would also like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported by NSERC, Google, Samsung, and ONR Grant N00014-14-1-0232. 8

2 References

[1] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In ICML, pages 160?167, 2008. [2] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In ACL, pages 384?394, 2010. [3] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In EMNLP, pages 1631?1642, 2013. [4] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In NIPS, pages 3111?3119, 2013. [5] Phil Blunsom, Edward Grefenstette, Nal Kalchbrenner, et al. A convolutional neural network for modelling sentences. In ACL, 2014. [6] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. ICML, 2014. [7] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In NIPS, pages 2787?2795, 2013. [8] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In NIPS, pages 926?934, 2013. [9] Yann N Dauphin, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. Zero-shot learning for semantic utterance classification. ICLR, 2014. [10] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeffrey Dean, and

Tomas Mikolov MarcAurelio Ranzato. Devise: A deep visual-semantic embedding model. NIPS, 2013. [11] Graham W Taylor and Geoffrey E Hinton. Factored conditional restricted boltzmann machines for modeling motion style. In ICML, pages 1025?1032, 2009. [12] Ryan Kiros, Richard S Zemel, and Ruslan Salakhutdinov. Multimodal neural language models. ICML, 2014. [13] Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In ICML, pages 1017?1024, 2011. [14] Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In ICML, pages 641?648, 2007. [15] Roland Memisevic and Geoffrey Hinton. Unsupervised learning of image transformations. In CVPR, pages 1?8, 2007. [16] Alex Krizhevsky, Geoffrey E Hinton, et al. Factored 3-way restricted boltzmann machines for modeling natural images. In AISTATS, pages 621?628, 2010. [17] Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. Semisupervised recursive autoencoders for predicting sentiment distributions. In EMNLP, pages 151?161, 2011. [18] Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. Semantic compositionality through recursive matrix-vector spaces. In EMNLP, pages 1201?1211, 2012. [19] Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. Inducing crosslingual distributed representations of words. In COLING, pages 1459?1474, 2012. [20] Sarath Chandar A P, Stanislas Lauly, Hugo Larochelle, Mitesh M Khapra, Balaraman Ravindran, Vikas Raykar, and Amrita Saha. An autoencoder approach to learning bilingual word representations. NIPS, 2014. [21] Karl Moritz Hermann and Phil Blunsom. Multilingual distributed representations without word alignment. ICLR, 2014. [22] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In ACL, pages 115?124, 2005. [23] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In MT summit, volume 5, pages 79?86, 2005. [24] Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. Effects of age and gender on blogging. In AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs, volume 6, pages 199?205, 2006. [25] Laurent Charlin, Richard S Zemel, and Craig Boutilier. A framework for optimizing paper matching. UAI, 2011. [26] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. KDD, 2014.

9