Learning Structural Correspondences Across Different Linguistic Domains with Synchronous Neural Language Models

Stephan Gouws and **GJ van Rooyen**
MIH Medialab, Stellenbosch University
SOUTH AFRICA
{stephan,gvrooyen}@ml.sun.ac.za

Yoshua Bengio
Dept. IRO, Université de Montréal
CANADA
bengioy@iro.umontreal.ca

Abstract

We introduce a novel framework for learning structural correspondences between two linguistic domains based on training synchronous neural language models with co-regularization on both domains simultaneously. We show positive preliminary results indicating that our framework can be successfully used to learn similar feature representations for correlated objects across different domains, and may therefore be a successful approach for transfer learning across different linguistic domains.

1 Introduction

Statistical discriminative methods for natural language processing (NLP) learn to combine features in the textual domain of interest, to predict the labels that words or phrases belong to (e.g. NOUN, VERB, etc. for part-of-speech tagging, and PERSON, ORGANISATION, etc. for information extraction). However, the performance of these models degrades quickly when they are presented with text drawn from a different domain which does not resemble the original training distribution. The process of porting a statistical model trained on one domain to perform well on another is called transfer learning, or domain adaptation in the NLP literature.

The ideas presented here are inspired by Structural Correspondence Learning (SCL) [6], a technique for domain adaptation which extends that of Ando & Zhang [1] and which aims to find the commonalities between the feature spaces of two different domains. SCL makes use of unlabelled target data to find common features between the two domains, thereby reducing their differences from a statistical point of view. The technique makes use of so-called "pivot features", features constructed on the individual original feature spaces and which behave similarly for discriminative learning in both domains. It is a linear technique which finds a common lower-dimensional projection of the pivot features such that labels available for the one domain may be projected via this common feature representation to be useful in the other domain.

Another major source of inspiration are Neural Language Models (NLMs) [3] which *jointly* learn unsupervised real-valued word feature vectors (called word *embeddings*) with the other model parameters. These word embeddings capture the statistical structure of the domain they are trained on, so that semantically similar words tend to have a nearby embedding (similar feature representation). Word representations learned using NLMs have previously been shown to be effective for intermediate-level NLP tasks, such as chunking and extracting named-entities [8, 11].

One crucial issue with SCL is how to select the pivot features, and it usually remains difficult and domain-dependent [2]. Another issue is with the assumption of linear correspondence between the feature spaces of the two domains via the dependence on the linear dimensionality reduction step. Our work builds on the idea of finding feature correspondences between two domains, but

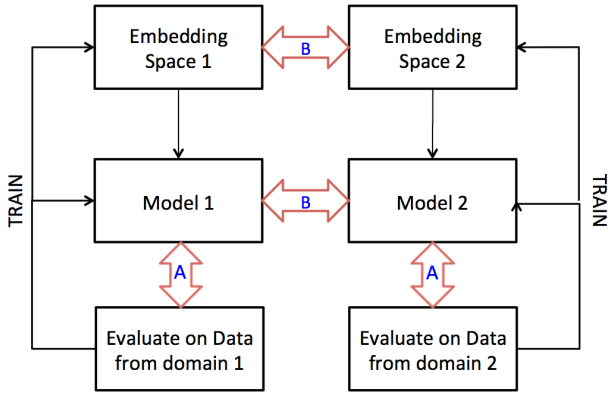

Figure 1: Overview of deep structural correspondence learning. Bold arrows between embeddings and between models represent constraints on the learned vector representations and NLM functions respectively. A represents the negative log-likelihood constraint $J^{nll}(\theta_\mathbf{i})$, and B the similarity constraints $J^R(\theta_\mathbf{i}, \theta_\mathbf{j})$ and $J^f(\theta_\mathbf{i}, \theta_\mathbf{j})$ respectively.

we relax these two assumptions: First, we make use of the ability of NLMs to learn fully unsupervised word representations in each domain, potentially trading training time for human feature-engineering time. Second, we make no assumption on the nature of the mapping between input feature spaces and shared embedding spaces, but rather let the networks learn this during training.

In this work we present a framework for transfer learning across different linguistic domains consisting of training NLMs synchronously *on multiple domains*. We investigate the hypothesis that two NLMs trained synchronously on two different domains can be used to learn similar feature representations for correlated objects (words) between the two domains, given sufficient co-regularization in terms of priors on the learned representations and on the model parameters themselves. Our framework provides a nonlinear analog to the linear Structural Correspondence Learning framework, and therefore in future work we propose to apply it to domain adaptation of feature-based statistical models based on learning embeddings of different objects, when the same (or closely related) objects can be found in both domains.

2 Deep structural correspondence learning framework

NLMs represent each word in the language as a *distributed* feature vector[1], and learns a *function* to combine vector representations of observed previous words into representations for the predicted target words. Each word maps to some point in a K-dimensional feature space, where each dimension corresponds to one such feature. Functionally similar words usually tend to cluster closer together along some dimensions in this space. This is the result of training on large volumes of text: syntactically or semantically similar words in similar contexts map to the same target words. A network trained with adequate regularization prefers to learn compact functions and pulls the representations of functionally similar words closer together to reduce its own modelling burden (satisfies the regularization constraint) while it optimizes for training set modelling accuracy (satisfying the negative log-likelihood constraint). *We would like to encourage this behaviour of learning similar representations for similar elements across more than one domain.*

An overview of our framework is depicted in Fig. 1. Deep structural correspondence learning consists of synchronously training two NLMs on two different domains, with strong priors on the induced embeddings and the learned functions of each model, so as to encourage common features between the domains to be extracted. More specifically, given two models m_i and m_j, we optimize a global cost function $J_{\text{global}} = \frac{1}{2}(J_i + J_j)$ consisting of the terms (only shown for m_i):

$$J_i = J^{\text{nll}}(\theta_\mathbf{i}) + \alpha J^R(\theta_\mathbf{i}, \theta_\mathbf{j}) + \beta J^f(\theta_\mathbf{i}, \theta_\mathbf{j}), \tag{1}$$

[1] As opposed to a *distributional* feature vector learned using for instance Brown clustering [7].

where $\theta_{\mathbf{i}}$ represents the parameters of model m_i, the first term represents m_i's negative-log-likelihood on its training set, and the second term (weighted by the coefficient α) represents the prior on the similarity between m_i and m_j's learned embeddings (therefore a function of $\theta_{\mathbf{i}}$ and $\theta_{\mathbf{j}}$), while the last term (weighted by the coefficient β) represents the prior that the models should represent similar functions (therefore also a function of both $\theta_{\mathbf{i}}$ and $\theta_{\mathbf{j}}$).

Informally, this global training objective tries to simultaneously satisfy two divergent criteria: Make the models and their learned features as similar as possible (the J^R and J^f terms), while modelling the language in each domain in the best possible way (the J^{nll} term). We hypothesize that the solution to this optimization problem yields an alignment in embedding space which captures the common latent features between the two domains. By providing the models with an initial seed list of words known to be similar in the two domains, the models can use the initial words as "pivots" or "anchors" to iteratively align (pull closer to each other) the distributionally similar words in the two vocabularies with respect to the pivots, and by extension with respect to each other.

$J^R(\theta_{\mathbf{i}}, \theta_{\mathbf{j}})$ in Eqn. 1 encodes our prior knowledge of the similarity between the vocabularies of the two domains. We want to provide the networks with a seed list of known distributionally similar words to start from. There are at least two possible ways to model this prior knowledge: The first is to add hard constraints on the learned representations by using the same underlying physical embedding vector $\mathbf{r}_i$ for common word w_i in both domains (known as "tied weights"). The second approach is to minimize a weighted distance term between known (or suspected) common words. In this work we only focus on the first approach and leave the second approach for future work.

The NLM consists of a learned function which models the learned regularities for combining words in the training language into new words, i.e. the syntax of the language, represented by the $J^f(\theta_{\mathbf{i}}, \theta_{\mathbf{j}})$ term. This term encodes our prior knowledge of the similarity between the syntactic structures over the different linguistic domains. To compute the similarity between the learned functions one cannot simply compute the distance between the weight vectors of the individual networks, since a network can permute its hidden units and their associated weights and still learn similar function mappings. We approximate this term by computing the distance between the outputs produced by the two networks *given the same input* [9]. I.e. given a common input minibatch of M training inputs, we compute the Euclidian distance between the vector formed for each model by concatenating all M predicted next word embedding vectors on the common input.

Finally, it has previously been established that the order in which training examples are presented to (especially) non-convex learning algorithms has an effect on the learning dynamics of the model [4], called *curriculum learning*. In that work, for language modelling, the strategy was to initially favour more frequently occurring words, and then gradually introduce less frequent words. We hypothesize that a similar curriculum strategy should improve the training convergence rate of pulling representations of similar words closer, since presenting both networks initially with examples favouring common word pairs allows the networks to learn informative representations for these pivot words with respect to one another. We hypothesize that this should make it easier to "slot in" new words later on. Our strategy is to partition our training set into decreasing number of common pivot words per training pair (a (context, target word) pair). Each group in the partition is subsequently sorted in decreasing order of mean frequency of occurrence of its words in the corpus. We therefore favour common words between the two domains primarily, and more frequent words in each domain secondarily.

3 Experiments and results

Our primary hypothesis is that two NLMs trained synchronously on two different domains can be used to learn similar feature representations for correlated objects (words) between the two domains, given sufficient co-regularization in terms of priors on the learned representations and on the model parameters themselves. In order to directly test this hypothesis, we create a controlled dataset where we can control the actual distribution of known distributionally similar word pairs.

We randomly sampled 1M consecutive words from the LA Times dataset. Tokens were lower-cased, and digits were conflated to skeleton representations[2]. We extracted the top-20K tokens (call this V) from this processed and tokenized dataset and then proceeded to create two distinct datasets as

[2]I.e. '342'→'###', and '1,033'→'####'

follows: Given a degree-of-overlap parameter k, we extract the top $k\%$ tokens V_c from V. For each remaining word $w_i \in V \backslash V_c$ [3], we create two distinct vocabularies V_1 and V_2 consisting of the tokens w_i + "_1" and w_i + "_2" respectively. We then encode two *disjoint* datasets (i.e. different, non-overlapping sections from the same training set), D_1 and D_2 using the vocabularies V_1 and V_2 respectively, each therefore containing k % overlap in terms of the original vocabulary V.

We can exploit the fact that we know which word pairs are similar in the original dataset to directly measure the effectiveness of the proposed methodology to learn common word representations, by simply measuring the average frequency-weighted distance between these known similar word-pair embeddings: Given a list of N known similar word pairs[4] (w_i, w_j), we compute the average word-pair distance as $\sqrt{\sum f_i(\mathbf{r}_i - \mathbf{r}_j)^2}$ where f_i is the normalized frequency of occurrence of word w_i. Importantly, we *measure* this distance for evaluation, but do not directly optimize for it.

It is also important to be aware of the general shrinking effect that any L_2 regularization terms may have on the average *magnitude* of the total embedding space, and not to confound such regularization-based shrinking of the embedding norms with the model actually learning to pair distributionally-similar word pairs. We therefore enforce a magnitude normalization step after each update to the embeddings as in Collobert [8]. However, instead of normalizing all embeddings individually to lie on the unit hyperspere, we normalize each embedding vector by the average magnitude over all embeddings, i.e. we maintain an average embedding vector norm of 1.

We constructed a dataset by choosing $k = 1$ % vocabulary overlap between the domains (200 word pairs), using a vocabulary of the 20,000 most frequent words in the corpus, resulting in a joint vocabulary of 39,800 words. We ran several experiments to evaluate our primary hypothesis, and also to evaluate the impact that (i) imposing a function-distance constraint, and (ii) employing a curriculum-based training strategy has on the average distance between known distributionally similar word pair embeddings.

We used our own Theano [5] implementation of the Log Bilinear NLM [10]. All models were trained with an adaptive learning rate starting at 0.1, decreasing slowly in the number of minibatches. Fig. 2 shows the average distances between the learned vector representations for the similar word pairs (lower is better) normalized to lie in the same unit interval for the 3 test cases that we evaluate, as training progresses. We see that without constraining the learned functions (setting $\beta = 0$ in the βJ^f term in Eqn. 1), the learned representations for similar word pairs *diverge*. However, with adequate constraints on the learned functions (a higher β value), the representations learned for similar word pairs are shown to start converging, which confirms our primary hypothesis. Interestingly, we found that imposing a curriculum ordering on the training data (not shown) leads to faster initial convergence rates on the curriculum boundaries, but does not lead to a faster convergence rate in general.

4 Conclusion

We introduced a novel framework for learning structural correspondences between two linguistic feature domains based on training synchronous neural language models (NLMs) with co-regularization *on multiple domains simultaneously*. We exploit the fact that NLMs learn unsupervised feature representations of the objects in the domains on which they are trained, and constrain the models to learn similar representations for objects which show correlated behaviour in the different training domains.

Our initial results are positive, and indicate that features learned for common objects do indeed become progressively more similar, which validates our hypothesis. In future work, we intend to apply this work to the transfer learning setting, to transfer feature-based sequence tagging models from one domain of text to another with minimal user intervention.

[3] Backslash indicates the set deletion operator.

[4] E.g. ('president_1', 'president_2').

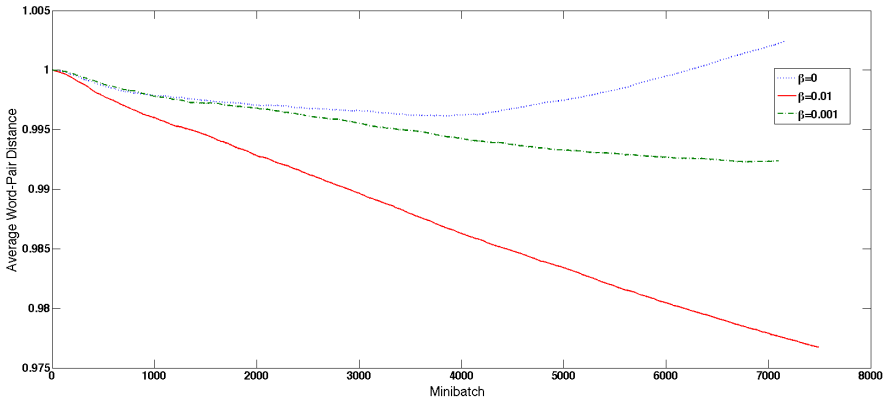

Figure 2: Average normalized distance between learned vector representations for similar word pairs across two different domains as training progresses over 2 epochs of the data (lower is better). The top solid line shows that training without constraining the learned functions of the neural language models leads to a divergence. The middle dashed line shows the positive effect of adding a weak prior that learned neural language models should be as similar as possible. Finally, the bottom solid line shows that increasing this prior leads to faster convergence.

References

[1] R.K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6:1817–1853, 2005.

[2] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems 20*, Cambridge, MA, 2007. MIT Press.

[3] Y Bengio, R Ducharme, and P Vincent. A neural probabilistic language model. *Journal of Machine Learning Research*, 2003.

[4] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[5] J Bergstra, O Breuleux, F Bastien, P Lamblin, R Pascanu, G Desjardins, J Turian, D Warde-Farley, and Y Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[6] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia, 2006.

[7] P.F. Brown, P.V. Desouza, R.L. Mercer, V.J.D. Pietra, and J.C. Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

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

[9] D Erhan, P Manzagol, Y Bengio, S Bengio, and P Vincent. The difficulty of training deep architectures and the effect of unsupervised pre-training. In *Journal of Machine Learning Research*, pages 153–160, April 2009.

[10] A. Mnih and G. Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM, 2007.

[11] J. Turian, L. Ratinov, and Y. Bengio. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.