A Margin-based Loss with Synthetic Negative Samples for Continuous-output Machine Translation

Gayatri Bhat **Sachin Kumar** **Yulia Tsvetkov**
Language Technologies Institute
Carnegie Mellon University
{gbhat,sachink,ytsvetko}@cs.cmu.edu

Abstract

Neural models that eliminate the softmax bottleneck by generating word embeddings (rather than multinomial distributions over a vocabulary) attain faster training with fewer learnable parameters. These models are currently trained by maximizing densities of pretrained target embeddings under von Mises-Fisher distributions parameterized by corresponding model-predicted embeddings. This work explores the utility of margin-based loss functions in optimizing such models. We present **syn-margin** loss, a novel margin-based loss that uses a *synthetic* negative sample constructed from only the predicted and target embeddings at every step. The loss is efficient to compute, and we use a geometric analysis to argue that it is more consistent and interpretable than other margin-based losses. Empirically, we find that syn-margin provides small but significant improvements over both vMF and standard margin-based losses in continuous-output neural machine translation.

1 Introduction

A new approach to conditional language modeling (Kumar and Tsvetkov, 2019) generates continuous-valued embeddings in place of discrete tokens (such as words or subwords). These embeddings are trained to lie in a pretrained word embedding space by maximizing, at each step of training, the von Mises-Fisher (vMF) probability density of the target pretrained embedding given the model-predicted embedding (§2). This eliminates the softmax bottleneck to ensure time- and memory-efficient training.

We investigate alternative loss functions for this new class of models, specifically *margin-based* formulations. These have been used to train embeddings for a range of tasks (Bojanowski et al., 2017; Bredin, 2017; Wu et al., 2018), and standard margin-based losses yield slight but inconsistent improvements over vMF on continuous-output neural machine translation (NMT). We propose **syn-margin**, a novel margin-based loss for which negative samples are *synthesized* using only the predicted and target embeddings, without sampling from or searching through the large pretrained embedding space (§3). These samples are constructed by extracting the portion of the predicted embedding that is not along the target embedding; intuitively, suppressing this component will increase the predicted embedding's similarity to the target. We use a geometric analysis to argue that this principled construction renders syn-margin loss more consistent and interpretable than standard margin-based losses that select negative samples randomly or heuristically (Collobert et al., 2011; Hadsell et al., 2006; Schroff et al., 2015; Mikolov et al., 2013a). Empirically, we find that syn-margin attains small but statistically significant improvements over vMF (§4) on continuous-output neural machine translation (NMT).

The key contributions of this work are: (1) the formulation of syn-margin loss, which is applicable across natural language processing and computer vision tasks for which the targets lie in pretrained embedding spaces (2) a geometric analysis of the functionality of syn-margin loss, which provides insights into the mechanism of margin-based losses in general and (3) the empirical result of improved performance on continuous-output NMT.

2 Continuous-output models

Conditional language models generate text conditioned on some input, e.g., produce translations of input sentences (Sutskever et al., 2014; Bahdanau et al., 2015; Luong et al., 2015). State-of-the-art

Proceedings of the 3rd Workshop on Neural Generation and Translation (WNGT 2019), pages 199–205
Hong Kong, China, November 4, 2019.
www.aclweb.org/anthology/D19-56%2d

neural models generate the text as a sequence of discrete tokens such as words.[1]

A traditional model, at every step of generation, produces a context vector $\mathbf{c}$ that encodes both the conditioning input and the output from previous generation steps. It then transforms $\mathbf{c}$ into a *discrete* distribution over the target vocabulary V using a softmax-activated linear transformation of size $|\mathbf{c}| \times |V|$. These models are typically trained using cross-entropy loss, and inference uses either greedy or beam decoding.

Instead of the multinomial distribution over V, continuous-output conditional language models generate a d-dimensional word embedding $\hat{\mathbf{u}}$ (Kumar and Tsvetkov, 2019). For this purpose, the $|\mathbf{c}| \times |V|$ transformation is replaced by a linear layer of size $|\mathbf{c}| \times d$. This design enables the model to have far fewer parameters than the original ($d \ll V$).

The model is trained and decoded in conjunction with a table of pretrained embeddings for words in V. Proposing that the *predicted embedding* $\hat{\mathbf{u}}$ parametrizes a von Mises-Fisher distribution over all d-dimensional vectors, Kumar and Tsvetkov (2019) train the model by maximizing the probability density at the target word's pretrained embedding $\mathbf{u}$ under this distribution centered at $\hat{\mathbf{u}}$:

$$p(\mathbf{u}; \hat{\mathbf{u}}) = C_m(\|\hat{\mathbf{u}}\|)e^{\hat{\mathbf{u}}^T\mathbf{u}}$$

where

$$C_m(\|\hat{\mathbf{u}}\|) = \frac{\|\hat{\mathbf{u}}\|^{d/2-1}}{(2\pi)^{d/2} I_{d/2-1}(\|\hat{\mathbf{u}}\|)}$$

with I_v being the modified Bessel function of the first kind of order v.

Thus, every predicted embedding is driven towards its *target embedding* $\mathbf{u}$, which can be identified since target words are available during training. This is much faster than training in the discrete-output case, since vMF densities are implicitly normalized. During inference, choosing the most likely word reduces to finding the predicted embedding's nearest neighbour (by cosine similarity) in the L_2-normalized pretrained embedding space.[2]

[1]The discrete units may be words, sub-word units (Sennrich et al., 2016), characters (Ling et al., 2015; Kim et al., 2016) or tokens of any other granularity. We focus on the generation of words, since pretrained embeddings spaces at this granularity are interpretable and semantically coherent across languages.

[2]In line with the inference mechanism, all references we make to 'similarity' or 'closeness' will be in the cosine, not Euclidean sense. In a slight change of notation, we will henceforth use $\hat{\mathbf{u}}$ to refer to the *unit vector* along a predicted embedding rather than the predicted embedding itself.

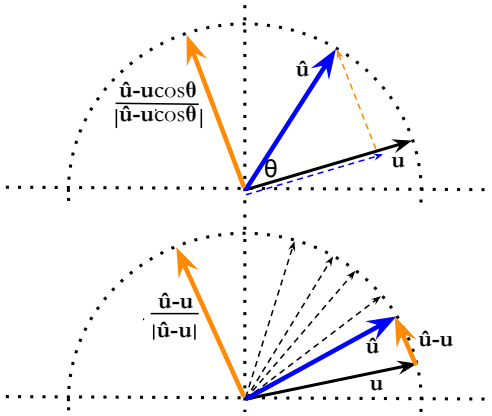

Figure 1: **Synthesizing negative samples.** Consider predicted and target embeddings $\hat{\mathbf{u}}$ (blue) and $\mathbf{u}$ (solid black), respectively. To synthesize a negative example for the margin-based loss by **projection** (top), we project $\hat{\mathbf{u}}$ onto $\mathbf{u}$ (dotted blue), use this to find component of $\hat{\mathbf{u}}$ that is orthogonal to $\mathbf{u}$ (dotted orange) and normalize it to obtain $\mathbf{u}_{\text{orth}}$ (solid orange). To synthesize by **difference** (bottom), we normalize $\hat{\mathbf{u}} - \mathbf{u}$ to obtain $\mathbf{u}_{\text{diff}}$ (long orange).

3 Margin-based loss formulations

To explore the space of additional loss functions for continuous-output models, we study margin-based losses commonly used to compare embeddings in both natural language and image processing (Collobert et al., 2011; Schroff et al., 2015) tasks:

$$\mathcal{L} = \max\{0, \lambda + \mathbf{u'}^T\hat{\mathbf{u}} - \mathbf{u}^T\hat{\mathbf{u}}\} \quad (1)$$

This requires $\hat{\mathbf{u}}$ to be closer to $\mathbf{u}$ than to some *negative sample* $\mathbf{u}'$ by a margin of λ. (Since the embeddings are normalized, inner product corresponds to cosine similarity.) Here λ is a hyperparameter that, along with $\mathbf{u}'$, decides whether $\hat{\mathbf{u}}$ is 'close enough' to $\mathbf{u}$.

The negative sample $\mathbf{u}'$ is usually chosen by (1) stochastic processes such as **negative sampling**: randomly choosing an embedding from the pretrained embedding table, and averaging loss over k random draws or (2) heuristic selections such as the **most informative negative sample** introduced by Lazaridou et al. (2015): the embedding in the table that is closest to $\hat{\mathbf{u}} - \mathbf{u}$.

3.1 The role of negative samples

What is the role of the negative sample in this margin-based loss? We investigate with a geomet-

ric analysis.

At the outset, consider predicted and target embeddings $\hat{\mathbf{u}}$ and $\mathbf{u}$, both of unit length. The predicted embedding's components *parallel* and *orthogonal* to $\mathbf{u}$ are $(\hat{\mathbf{u}}^T\mathbf{u})\mathbf{u}$ and $\hat{\mathbf{u}}-(\hat{\mathbf{u}}^T\mathbf{u})\mathbf{u}$ (dotted blue and dotted orange lines respectively in Figure 1, which illustrates this decomposition). Let the unit vector along this orthogonal component be $\mathbf{u}_{\text{orth}}$ (solid orange line). It follows that (1) $\mathbf{u}_{\text{orth}} \perp \mathbf{u}$ and (2) $\hat{\mathbf{u}}$ is a linear combination of these orthogonal vectors, say, $\lambda_1\mathbf{u} + \lambda_2\mathbf{u}_{\text{orth}}$.

Now, choose any embedding $\mathbf{x}$ of unit length from the d-dimensional space (not necessarily the pretrained embedding of any word) to use as the negative sample in a margin-based loss. Let its projections along $\mathbf{u}$ and $\mathbf{u}_{\text{orth}}$ be $\lambda_3\mathbf{u}$ and $\lambda_4\mathbf{u}_{\text{orth}}$ Since these are orthogonal, $\mathbf{x}$ decomposes as $\lambda_3\mathbf{u} + \lambda_4\mathbf{u}_{\text{orth}} + \mathbf{y}$ where $\mathbf{y}$ is some vector orthogonal to both $\mathbf{u}$ and $\mathbf{u}_{\text{orth}}$ ($\mathbf{y} = \mathbf{0}$ when $d = 2$).

Using the decomposed forms of $\hat{\mathbf{u}}$ and $\mathbf{u}_{\text{orth}}$ in the margin-based loss, the second argument of equation 1 becomes

$$\lambda+\lambda_4{\mathbf{u}_{\text{orth}}}^T\hat{\mathbf{u}}-(1-\lambda_3)\mathbf{u}^T\hat{\mathbf{u}}+\mathbf{y}^T(\lambda_1\mathbf{u}+\lambda_2\mathbf{u}_{\text{orth}})$$

Applying orthogonality to set the final term to zero gives

$$\mathcal{L} = \max\{0, \lambda + \lambda_4{\mathbf{u}_{\text{orth}}}^T\hat{\mathbf{u}} - (1 - \lambda_3)\mathbf{u}^T\hat{\mathbf{u}}\}$$

Thus, *regardless of the actual negative sample chosen*, the loss reduces to a form wherein some scalar multiples of $\mathbf{u}$ and $\mathbf{u}_{\text{orth}}$ are the positive and negative samples respectively. The loss essentially penalizes the component of $\hat{\mathbf{u}}$ that is orthogonal to $\mathbf{u}$.

3.2 Synthesized negative samples

Drawing on this insight, we propose to use the *synthesized* vector $\mathbf{u}_{\text{orth}}$ as the negative sample in a margin-based loss. This sets λ_4 and λ_3 at 1 and 0 respectively, providing a steady training signal. In contrast, these coefficients fluctuate during training if heuristic or stochastic methods are used to select negative samples. We also propose a second closely related negative sample $\mathbf{u}_{\text{diff}}$, synthesized by subtraction rather than projection: the unit vector along the difference $\hat{\mathbf{u}} - \mathbf{u}$ (see Figure 1 for a visualization). Synthesizing $\mathbf{u}_{\text{orth}}$ and $\mathbf{u}_{\text{diff}}$ is efficient since it does not require any sampling from or searching through the pretrained embedding space. We refer to the loss formulations using $\mathbf{u}_{\text{orth}}$ and $\mathbf{u}_{\text{diff}}$ as **syn-margin** by projection (SMP) and difference (SMD) respectively.

Although $\mathbf{u}_{\text{orth}}$ and $\mathbf{u}_{\text{diff}}$ are functions of $\hat{\mathbf{u}}$, they are plugged into $\mathcal{L}$ as *constant* vectors detached from the computational graph; this prevents them from being optimized to minimize $\mathcal{L}$. We highlight that using these synthesized negative samples cannot lead to a degenerate state in which all the word embeddings collapse to a single point. This is because the target embeddings are, unlike in some previous work that uses margin-based losses, pretrained and fixed.

4 Experimental Setup

We follow Kumar and Tsvetkov (2019) to conduct experiments on neural machine translation.

Datasets We evaluate our models on IWSLT'16 (Cettolo et al., 2015) French→English and German→English datasets. We pretrain target embeddings on a large English-language corpus (4B+ tokens) using FastText on default settings (Bojanowski et al., 2017) and L_2-normalize the embeddings. Vocabulary sizes are limited to 50000. We follow Kumar and Tsvetkov (2019) in using the standard development (tst2013 and tst2014) and test (tst2015 and tst2016) sets associated with the parallel corpora and in processing the data; train, development and test splits contain roughly 200K, 2300 and 2200 parallel sentences each.

Setup We use a neural machine translation system with attention (Bahdanau et al., 2015), set up to match that described in Kumar and Tsvetkov (2019). The encoder and decoder are 1-layer bidirectional and 2-layer LSTMs with 1024-dimensional hidden and output states. Word embeddings are 512-dimensional on the encoder side and 300-dimensional on the decoder side. Decoder input and target embeddings are tied to the same parameter matrix, these embeddings are transformed to the correct dimensions with a linear layer when used as inputs to the decoder. Generated embeddings are normalized before computing margin-based losses (vMF loss accounts separately for embedding norm). We train for up to 20 epochs with Adam (Kingma and Ba, 2015), an initial learning rate of 0.0005 and no dropout. During inference, vMF density is used to choose an output word given an embedding predicted by the vMF system, and the predicted embedding's nearest neighbour is chosen as the output for margin-trained systems. Hyperparameters are selected using performance on the development set and we report means and standard deviations of BLEU

Output Type	Loss Function	IWSLT Fr→En	IWSLT De→En
Discrete	**Cross-entropy: untied embeddings**	31.3 ± 0.4	25.1 ± 0.2
	Cross-entropy: tied embeddings	31.3 ± 0.9	24.8 ± 0.2
Continuous	**von Mises-Fisher**	31.8 ± 0.3	25.0 ± 0.2
	Most informative negative sample	32.0 ± 0.2	$25.1^{\ddagger} \pm 0.1$
	Negative sampling	$32.2^{*} \pm 0.4$	24.8 ± 0.2
	Syn-margin by difference (SMD)	$32.0^{*} \pm 0.3$	$\mathbf{25.4}^{*\dagger\ddagger} \pm 0.3$
	Syn-margin by projection (SMP)	$\mathbf{32.3}^{*\dagger} \pm 0.2$	$25.3^{*\ddagger} \pm 0.5$

Table 1: **Experimental results.** Means and standard deviations of BLEU scores across 4 runs of each experiment, for the (1) discrete-output baseline, (2) continuous-output models trained using vMF, most informative negative example (Lazaridou et al., 2015) and negative sampling, and (3) proposed syn-margin losses constructed using vector projection and vector difference, on IWSLT'16 Fr→En and De→En datasets. Asterisks, daggers and double daggers indicate significant gains over vMF, most informative negative sample and negative sampling respectively ($p = 0.05$).

scores (Papineni et al., 2002) over 4 runs of each experiment.

Baselines and benchmarks We compare syn-margin losses constructed using projection (SMP) and difference (SMD) techniques against: (1) vMF loss (specifically, the negative log-likelihood formulation in the original paper), (2) margin-based loss averaged over 5 negative samples drawn uniformly at random from the pretrained word embeddings and (3) margin-based loss using the most informative negative sample (Lazaridou et al., 2015). We also report results on a softmax-based system with identical architecture except in the last layer, initializing the softmax parameters with pretrained embeddings, with and without tied embeddings.

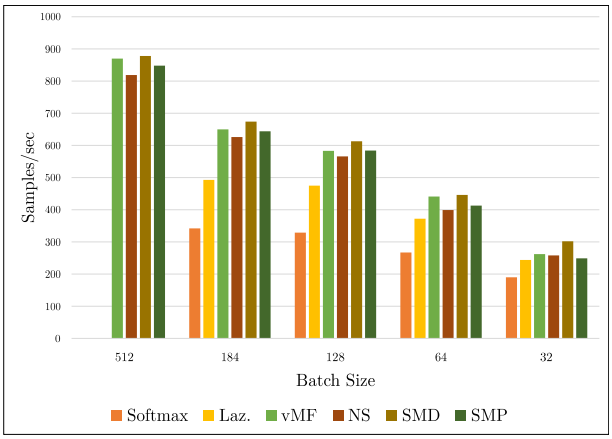

Figure 2: **Speed comparisons.** We compare the number of training instances that can processed per second for each loss formulation. Syn-margin is found to be faster than other margin-based methods, and comparable in speed to vMF.

5 Results and Analysis

Syn-margin methods show small (+0.4 and +0.5 BLEU) and statistically significant gains over vMF on both datasets, although there is no consistent winner among the two syn-margin variants (Table 1). The improvement over most informative negative sample and negative sampling is less prominent, and significant only in some cases. Syn-margin's computational efficiency matches that of vMF (Figure 2).

Comparing translations produced by vMF and syn-margin models in the Fr→En task, we find SMP translations to be more grammatical. They better preserve grammatical information such as gender (SMP correctly predicts the fragment 'her personality' while vMF generates 'his personality') and tense (SMP generates 'does it predict' while vMF produces 'does it predicted'), and are better-formed without cascading errors.

Next, to develop a qualitative understanding of the synthesized negative samples, we identify predicted embeddings' and SMP negative samples' nearest neighbours (NN) among the pretrained target embeddings. Either both embeddings share a common NN, or in a weak pattern, the SMP's NN captures $\hat{\mathbf{u}}$'s semantic or grammatical divergence from $\mathbf{u}$. For instance, where the target is 'means' and the prediction's NN is 'meant', the negative sample's NN 'held' penalizes past-tense information in the predicted embedding. Similarly, target 'Hollywood' and prediction 'movies' are associated with negative sample 'concerts'. This confirms our intuition about the functionality of

	vMF	Laz	Random	SMD	SMP
Correct prediction: similarity to nearest neighbour	**0.96**	0.87	0.88	0.91	0.88
Wrong prediction: similarity to nearest neighbour	**0.91**	0.80	0.83	0.88	0.86
Wrong prediction: similarity to target embedding	0.39	0.28	0.21	0.41	**0.42**
Accuracy (%)	23.55	23.18	23.39	23.89	**24**
Accuracy @2 (%)	28.91	28.04	28.28	29.59	**29.89**
Accuracy @5 (%)	32.23	31.45	31.16	32.96	**33.22**
Accuracy @10 (%)	34.77	33.85	33.42	35.49	**35.6**

Table 2: **Error margins and accuracies.** The average similarity of predicted embeddings to their nearest neighbours is lower in SMP/SMD-trained models than in vMF-trained models. Among predicted embeddings whose nearest neighbours are not the targets, similarity to the targets increases when we switch from vMF to syn-margin loss. This is potentially linked to the increase in accuracies @2, 5 and 10 that results from the switch to syn-margin loss.

margin-based losses in general and syn-margin in particular.

We briefly analyze the properties of embeddings predicted by vMF and SMP Fr→En systems. Among incorrect predictions (cases in which the pretrained embedding closest to $\hat{\mathbf{u}}$ is not $\mathbf{u}$), the average cosine similarity between predicted embeddings and their nearest pretrained embeddings falls from vMF to SMP (0.91 to 0.86), while that between the predicted and target embeddings rises (0.39 to 0.42). This is accompanied by increases in accuracy @2, @5 and @10 (Table 2).

6 Related Work

Pretrained embeddings trained in an unsupervised manner (Mikolov et al., 2013a) are used as input and intermediate representations of data for natural language processing tasks such as part-of-speech tagging and named entity recognition (Ma and Hovy, 2016), sentiment analysis (Tang et al., 2016) and dependency parsing (He et al., 2018).

We build on (Kumar and Tsvetkov, 2019), one of the first instances of using pretrained embeddings as model *outputs* for complex sequence-generation tasks. Closely related work on embedding prediction includes zero-shot learning for word translation (Nakashole, 2018; Conneau et al., 2018) and image labeling (Lazaridou et al., 2015), as well as rare word prediction (Pinter et al., 2018) and classification (Card et al., 2019).

Margin-based losses are commonly used to train neural networks that predict dense vectors for classification tasks, and have long been used in computer vision. Standard formulations include contrastive (Hadsell et al., 2006) and triplet (Schroff et al., 2015) losses; triplet loss is identical to the max-margin framework we use. Other closely related approaches are the imposition of an angular margin constraint and the minimization of distance to the farthest intra-class example coupled with maximization of distance to the nearest inter-class example (Liu et al., 2016; Deng et al., 2017). In contrast to syn-margin, many of these losses pertain to *trainable* target embedding spaces.

The triplet loss has also been used in various NLP applications (Collobert et al., 2011). Techniques used to pick negative samples include perturbing training data (Smith and Eisner, 2005), sampling according to word frequency (Mikolov et al., 2013b), sampling until a non-zero loss is obtained (Weston et al., 2011) and searching for the negative sample that gives the largest (Rao et al., 2016) or most informative (Lazaridou et al., 2015) loss. These techniques also correspond to trainable target embedding spaces, and are all equally or less efficient than syn-margin.

7 Conclusion

We explore the use of margin-based loss functions to train continuous-output neural models, providing a geometric analysis of their functionality in this framework. Through this analysis, we develop a principled method to synthesize negative samples for margin-based losses, efficiently and on the fly. We argue that these negative samples are more consistent and interpretable than those picked using stochastic or heuristic techniques. Experiments on neural machine translation show that the proposed syn-margin loss improves over vMF and is either comparable or preferable to other margin-based losses. The analysis and loss function we propose are more generally applicable to neural models whose outputs lie in pretrained embedding spaces.

Acknowledgments

We gratefully acknowledge Anjalie Field, Aditi Chaudhury, Elizabeth Salesky, Shruti Rijhwani and our anonymous reviewers for the helpful feedback and discussions. This material is based upon work supported by NSF grant IIS1812327 and an Amazon MLRA award.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. ICLR*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*.

Hervé Bredin. 2017. Tristounet: Triplet loss for speaker turn embedding. In *Proc. ICASSP*.

Dallas Card, Michael Zhang, and Noah A. Smith. 2019. Deep weighted averaging classifiers. In *Proc. FAT**.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2015. The IWSLT 2015 evaluation campaign. In *Proc. IWSLT*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *Proc. ICLR*.

Jiankang Deng, Yuxiang Zhou, and Stefanos Zafeiriou. 2017. Marginal loss for deep face recognition. In *Proc. CVPR, Faces in-the-wild Workshop/Challenge*.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *Proc. CVPR*.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Unsupervised learning of syntactic structure with invertible neural projections. In *Proc. EMNLP*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Proc. AAAI*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. ICLR*.

Sachin Kumar and Yulia Tsvetkov. 2019. Von Mises-Fisher loss for training sequence to sequence models with continuous outputs. In *Proc. ICLR*.

Angeliki Lazaridou, Georgiana Dinu, and Marco Baroni. 2015. Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proc. ACL*.

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proc. EMNLP*.

Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. 2016. Large-margin softmax loss for convolutional neural networks. In *Proc. ICML*.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proc. EMNLP*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNS-CRF. In *Proc. ACL*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *ICLR Workshop*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proc. NeurIPS*.

Ndapa Nakashole. 2018. Norma: Neighborhood sensitive maps for multilingual word embeddings. In *Proc. EMNLP*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proc. ACL*.

Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. 2018. Mimicking word embeddings using subword RNNs. In *Proc. ACL*.

Jinfeng Rao, Hua He, and Jimmy Lin. 2016. Noise-contrastive estimation for answer selection with deep neural networks. In *Proc. CIKM*.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proc. CVPR*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. ACL*.

Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proc. ACL*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proc. NIPS*.

Duyu Tang, Furu Wei, Bing Qin, Nan Yang, Ting Liu, and Ming Zhou. 2016. Sentiment embeddings with applications to sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Proc. IJCAI*.

Ledell Yu Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. 2018. Starspace: Embed all the things! In *Proc. AAAI*.