Going Beyond Word Cooccurrences in Global Lexical Selection for Statistical Machine Translation using a Multilayer Perceptron

Alexandre Patry
KeaText
845 Blvd. Décarie, Suite 202
Montréal, Canada H4L 3L7
alexandre.patry@keatext.com

Philippe Langlais
DIRO/RALI
Université de Montréal
Montréal, Canada H3C 3J7
felipe@iro.umontreal.ca

Abstract

Phrase-based statistical machine translation (PBSMT) decoders translate source sentences one phrase at a time using strong independence assumptions over the source phrases. Translation table scores are typically independent of context, language model scores depend on a few words surrounding the target phrase and distortion models do not influence directly the choice of target phrases.

In this work, we propose to condition the selection of each target word on the whole source sentence using a multilayer perceptron (MLP). Our interest in MLP lies in their hidden layer which encodes source sentences in a representation that is not directly tied to the notion of word.

We evaluated our approach on an English to French translation task. Our MLP model was able to improve BLEU scores over a standard PBSMT system.

1 Introduction

Phrase-based statistical machine translation systems translate source sentences step by step, starting with an empty sentence and ending when all source words have been translated (Koehn et al., 2003). At each step, an untranslated phrase is selected and one of its translation is appended at the end of the translation.

In this work, we are interested in the selection of a target phrase to translate a given source phrase. This selection is usually guided by three families of models. Translation models evaluate the intrinsic quality of a given phrase pair using evidences such as cooccurrence statistics between source and target words or phrases. These models always compute the same scores for a given pair of phrases, wherever it is used. A second family are language models, which evaluate the likeliness of target n-grams[1] independently from the source sentence. A third family are distortion models, whose main purpose is to evaluate the likelihood of phrase reorderings.

All these models only consider a fraction of the information available at a time. Figure 1 presents a partial translation that could be encountered by a decoder where these fractions of information are not enough to make a good decision. In this example, the decoder has already translated all but the last source word and it must decide if *plant* is translated by *usine* (a building) or *plante* (the botanic sense). As the position of the source word is fixed, the distortion model will not be of any help. If the system uses a 3-gram, only the last two words (*de cette*) of the partial hypothesis will be considered. Finally, the translation model will score the two options considering only *plant*. Therefore, the word *leaves* is never considered and the final decision largely depends on whether the training corpus is more biased toward one translation or the other.

This problem is not new and has even been addressed by the creators of CANDIDE (Berger et al., 1994), one of the first SMT system. We know of three groups of approaches for tackling it.

A first group of approaches acknowledge the bias of the system and use it at its advantage by customizing the training corpus for each source sentence. This bias can be introduced with thematic training corpora (Xu et al., 2007; Lü et al., 2007) or with custom corpora built dynamically to

[1] Sequences of n words where n usually varies between 3 and 5.

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 658–666,
Chiang Mai, Thailand, November 8 – 13, 2011.

Source	~~the leaves of this~~ plant
Partial target	les feuilles de cette
Pair 1	plant → usine
Pair 2	plant → plante

Figure 1: An example state where the selection of a target phrase depends mostly on the bias of the training corpus (words already translated are strike through).

be similar to the source sentence (Hildebrand et al., 2005; Lü et al., 2007).

A second group of approaches cast the target phrase selection problem in a word sense disambiguation setting where source phrases are considered as ambiguous words and their translations as different meanings (Vickrey et al., 2005). This usually boils down to training one classifier per source phrase. These classifiers use a variety of features like surrounding source words, target words, part-of-speech or lemmas(Berger et al., 1994; Carpuat and Wu, 2007; Stroppa et al., 2007; Chan et al., 2007). Instead of training one classifier per source phrase, Gimpel and Smith (2008) use the same contextual scores for all the source phrases. The weights of those scores are then optimized with the other weights of the decoder.

Finally a third group of approaches assign a probability to every words in the target vocabulary given a source sentence (Venkatapathy and Bangalore, 2009; Mauser et al., 2009; Patry and Langlais, 2009). This predicted vocabulary can guide the decoder at translation time.

Our approach stands in this last group and we present its general idea in section 2. While previous approaches used linear or logistic regression models, we opted for the multilayer perceptron presented in section 3. Section 4 motivates this choice with a simple example. Section 5 details the algorithm to train our MLP. Experimental results, previous works and conclusion then follow in sections 6, 7 and 8 respectively.

2 Target vocabulary prediction

Standard PBSMT systems condition their translation on one source phrase at a time. In this work, we propose a new model conditioning its score on the complete source sentence. We treat the prediction of each target word as a Bernouilli trial where the presence of a word is a success and its absence a failure. The probability of a target sentence can thus be evaluated with:

$$\Pr(\mathbf{t} \mid \mathbf{s}) = \overbrace{\prod_{t \in \mathbf{t}} \Pr(t \mid \mathbf{s})}^{\text{Present}} \overbrace{\prod_{t \notin \mathcal{T} - \mathbf{t}} 1 - \Pr(t \mid \mathbf{s})}^{\text{Absent}} \quad (1)$$

where t is a target word, $\mathbf{t}$ a target sentence, $\mathcal{T}$ the target vocabulary and $\mathbf{s}$ the source sentence.

We are now left with the problem of evaluating $\Pr(t \mid \mathbf{s})$, the probability of a target word given a source sentence. Previous work have modelled this distribution with linear models like IBM Model 1 (Mauser et al., 2009):

$$\Pr_{\text{IBM1}}(t \mid \mathbf{s}) = \frac{1}{|\mathbf{s}|} \sum_{s \in \mathbf{s}} \Pr(t \mid s) \quad (2)$$

or a logistic regression model (Venkatapathy and Bangalore, 2009; Mauser et al., 2009):

$$\Pr_{lr}(t \mid \mathbf{s}) = \text{sigmoid}\left(\sum_{s \in \mathbf{s}} w_{t,s}\right) \quad (3)$$

$$\text{sigmoid}(z) = \frac{1}{1 + e^{-z}} \quad (4)$$

where $w_{t,s}$ is a weight between the tokens s and t.

Both models assign weights directly between source and target words. We opted instead for a multilayer perceptron where source and target words are connected indirectly through an hidden layer.

3 Multilayer perceptrons

Instead of assigning weights between source and target words, our MLP project the source words in an artificial representation offered by the hidden layer, and then project this artificial representation on the target vocabulary. The architecture of our MLP is as follow:

$$\vec{h} = \tanh(W\vec{s}) \quad (5)$$

$$\vec{y} = \text{tsigmoid}(V\vec{h}) \quad (6)$$

$$\text{tsigmoid}(z) = \text{sigmoid}(z - 4.6) \quad (7)$$

where V and W are weight matrices to optimize, $\vec{s}$ the source sentence encoded in an one-hot vector, $\vec{h}$ contains the values of the hidden units and $\vec{y}$ contains the prediction probabilities of all target words in $\mathcal{T}$.

We use a sigmoidal activation function on the output layer (eq. 6) because it returns values between 0 and 1 that can be interpreted as probabilities. Sigmoid returns 0.5 when its input is 0,

Source	Target
the floor of the plant	le plancher de l'usine
the stem of the plant	la tige de la plante
the leaves and stem of the flower	les feuilles et la tige de la fleur
the floors and walls of the house	les planchers et les murs de la maison

Figure 2: Training corpus for our motivating example.

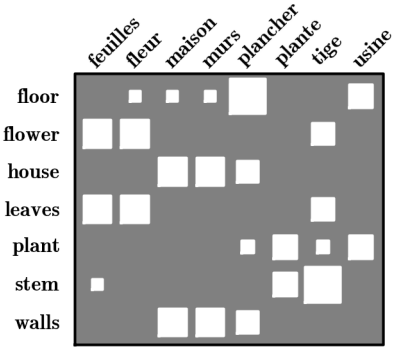

Figure 3: Weights learned for a logistic regression model optimized by gradient descent.

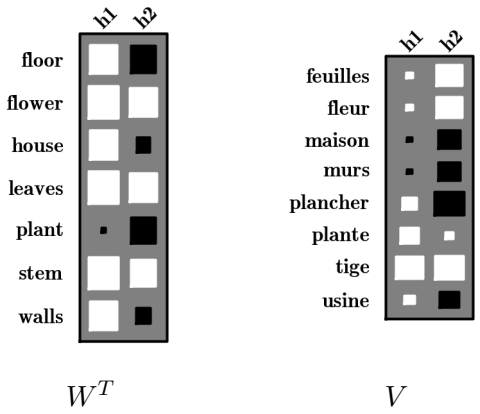

Figure 4: Weights learned for a MLP.

meaning that empty sentences encoded by vectors containing only zeros yield probabilities of 0.5 for all target word. It is clearly not what we want, so we use instead a modified sigmoid function (eq. 7) where empty sentences entail small probabilities for all target words (0.01 in our case). We could have used the same function for the hidden layer, but we opted for the hyperbolic tangent as it is known to shorten training time (LeCun et al., 1998).

More details on the training of our MLP are given in section 5.

4 Motivating Example

Conditioning the translation on the whole source sentence has already been studied. Previous approaches used linear or logistic regression models assigning a weight between every pair of source and target words. The number of connections in those models is thus quadratic in the size of the source and target vocabularies. Figure 3 shows the weight matrix of a logistic regression models trained by gradient descent on the sentences of Figure 2. The matrix is displayed in an Hinton diagram where white squares represent positive weights and black squares represent negative weights. The size of the squares are proportional to the absolute value of the weights.

Only pairs of words cooccurring in the training corpus get a weight different from zero, making the weight matrix sparse. This sparsity helps the model to scale well to larger vocabulary, but it cannot model relations in translation beyond word cooccurrences.

On the other hand, a MLP where source words are first projected into an artificial representation, and then on the target words, is not sparse. Figure 4 presents the weight matrices that were learnt from our training corpus. Any source word can thus contribute to the probability of any target word even when both words do not cooccur in the training corpus.

To show this fundamental difference between both kinds of models, we trained a logistic regression model and a MLP on the training corpus presented in Figure 2. We used these two models to predict the French words from the content words of *the leaves of the plant*. Both set of predictions are presented in Figure 5.

The regression model is unable to favour *plante* over *usine* because both words cooccur the same number of times with *plant* but do not cooccur with *leaves*. On the other hand, the MLP is able to favour *plante* because its artificial representation can separate words related to botanic (white squares in h_2 columns of Figure 4) from words related to buildings (black squares in h_2 columns of Figure 4).

Figure 6 shows the projection of four sentences on the artificial representation. Even if the four sentences share all the same words but one, the MLP is able to group sentences about botanic and

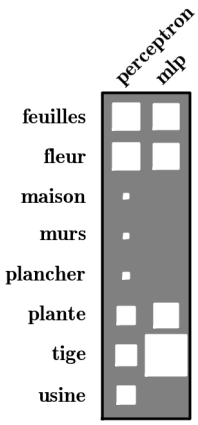

Figure 5: Predictions of the logistic regression model and the MLP for *the leaves of this plant*.

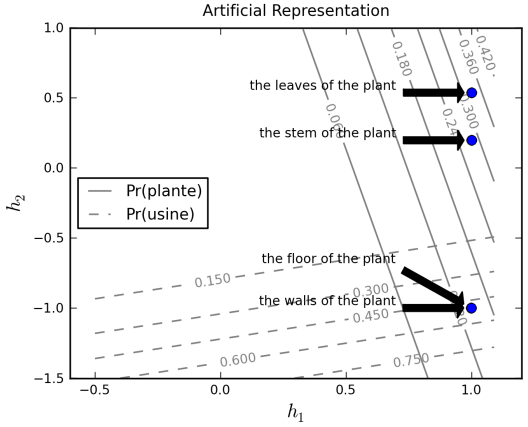

Figure 6: Projection of four sentences on the artificial representation.

buildings in different zones.

Leveraging the artificial representation offered by MLP to overcome data sparsity is not a novel idea: Bengio et al. (2003) and Schwenk (2007) trained state-of-the-art language models for speech recognition and machine translation using MLP.

5 MLP Training

To train our MLP, we must select values for V and W that will optimize the likelihood of our training data (eq. 1). We do so by minimizing the following error function using a gradient descent algorithm:

$$E(\mathbf{s}, \mathbf{t}) = \sum_{t \in \mathcal{T} - \mathbf{t}} \log\left(1 - y_{t|\mathbf{s}}\right) - \sum_{t \in \mathbf{t}} \log y_{t|\mathbf{s}} \quad (8)$$

1. Initialize V and W with $U(-0.05, 0.05)$.
2. For iterations 1 to 20:
   (a) For each batch of 100 sentences pairs:
      i. Compute the gradients of V and W with respect to eq. 8.
      ii. Test the following learning rates in orders 0.5, 0.05, 0.01 and select the first that decreases the error (eq. 9).
      iii. When such a learning rate exists, update V and W in the direction of their negative gradients multiplied by the learning rate.
3. Return V and W.

Figure 7: Gradient descent algorithm that we used to train our MLP.

where $y_{t|\mathbf{s}}$ is the probability that t appears in a translation of $\mathbf{s}$ according to our MLP. Gradient descent algorithms for MLP are already covered in many textbooks (Bishop, 1995), but because our models contain millions of weights, we had to develop some heuristics in order to keep the training time reasonable. Our modified gradient descent algorithm is presented in Figure 7.

This algorithm starts by initializing the values of V and W with a uniform between $(-0.05, 0.05)$. We varied the range of this uniform without notable variations in the results.

Computing gradients on the whole dataset before each update is too time consuming. Therefore, we computed the gradients on mini-batch of 100 sentence pairs instead. We could have used a stochastic gradient descent as well (mini-batches of size 1), but mini-batches are an opportunity to easily parallelize the training algorithm. We can encode all the source sentences of a mini-batch in a matrix where each sentence stands on a column, thus ending up with matrices-matrices multiplications instead of matrices-vectors multiplications in equations 5 and 6. This is desired as the former are faster to compute than the latter for modern linear algebra toolkit like ATLAS (Whaley and Petitet, 2005).

Once the gradients are computed, we must select a learning rate. We found this step to be critical to the success of our models. We first tried with a fixed learning rates, but the results were disappointing. We then implemented Brent line

search algorithm (Press et al., 1992, §10.2), but it was painfully slow. We finally ended up with the line search algorithm described on lines 2(a)ii and 2(a)iii. Even though this algorithm is simple, it did as well as Brent line search to optimize the log-likelihood in our experiments but it was faster.

We added a L1 regularizer to the error function during the line search to penalize big updates that bring small improvements:

$$E_{\text{L1}}(\mathbf{s},\mathbf{t}) = E(\mathbf{s},\mathbf{t}) + 0.1\left(\sum_{ki}|w_{ki}| + \sum_{jk}|v_{jk}|\right) \quad (9)$$

However, we did not used this regularized error to compute the gradients because it tends to push all weights toward zero at each update. This behaviour is not desired in our setting because only the weights associated to present source words are updated in W. Many words only appear in a couple of batches and their weights would all be pushed toward zero if we would include the regularizer when computing the gradients.

6 Experiments

We integrated our prediction models (see section 6.1) in an in house multi-stack phrase-based decoder which has performances similar to those of PHARAOH (Koehn, 2004a) when used in the same conditions.

We trained our phrase-table using TRAIN-FACTORED-MODEL.PERL, a script available with the MOSES PBSMT (Koehn et al., 2007). We optimized the weights of our log-linear model to maximize BLEU on our development set with the Nelder-Mead algorithm (Press et al., 1992) on n-best lists of 2000 sentences. To keep the decoding time reasonable, we limited the number of translations per source phrase to 30 and the number of hypotheses per stack to 50. We conducted our experiments with a trigram language model.

We are aware that our system could be enhanced in several ways. The distortion models embedded in MOSES are known to improve quality upon the translations produced by PHARAOH, and larger n-gram models, such as 5-gram models, might deliver as well slightly better results. This is left as future work. As will be discussed in section 6.5, our system performs comparably to other state-of-the-art systems tested in similar settings, and therefore, the gains we observed by integrating our prediction model into the decoder are representative.

6.1 Prediction scores

We investigated two ways (scores) to integrate our trained prediction models to the decoder.

As our MLP are trained to maximize the likelihood of the training data (eq. 1), it would be natural to add the likelihood score to the log-linear model optimized by the decoder. Likelihood is however heavy to compute because it sums over the complete target vocabulary. We followed Mauser et al. (2009) who suggested to use the *odd* score instead:

$$odd(\mathbf{t}|\mathbf{s}) = \prod_{t\in\mathbf{t}} \frac{y_{t|\mathbf{s}}}{1-y_{t|\mathbf{s}}} \quad (10)$$

An odd of x for a given word means that this word is x times more likely to be present in the translation than to be absent from it. An interesting property of this score is that it is proportional to eq. 1 once the source sentence is known.

We evaluated a second score that counts the number of target words with a probability higher than a given threshold α:

$$pred(\mathbf{t} \mid \mathbf{s}) = \left|\{t \in \mathbf{t} \mid y_{t|\mathbf{s}} > \alpha\}\right| \quad (11)$$

We selected the threshold to maximize the f-measure when the predicted words are compared against the reference translation of our development corpus, as suggested in (Patry and Langlais, 2009).

A convenient property of these scores is that they can be computed one target word at a time. Each time a new phrase is appended to a partial translation, the decoder can thus compute odd and $pred$ on this new phrase and update the partial translation score accordingly.

6.2 Models and data

We evaluated our system on a French-English translation task. We used corpora that were made available for the *Fourth Workshop on Machine Translation* (Callison-Burch et al., 2009). We trained our models on EUROPARL and the NEWS-COMMENTARY sections. We used TEST2007 as our development corpus, TEST2008 as our in-domain test corpus and NEWSTEST2009 as our out-of-domain test corpus.

We used those data to train three prediction models:

IBM1 An IBM1 model (eq. 2) trained with GIZA++ (Och and Ney, 2003).

PERCEPTRON A perceptron with a translated sigmoidal activation function:

$$\Pr_{\text{PERCEPTRON}}(t \mid \mathbf{s}) = \text{tsigmoid}(U\vec{s}) \quad (12)$$

where U is a sparse matrix linking each source word with its 10 best translations according to IBM1.[2] This model is equivalent to logistic regression (eq. 12).

MLP-64 A MLP with 64 hidden units. This number of hidden units was selected after informal experiments on the development corpus.

Both MLP-64 and PERCEPTRON were trained using the algorithm of Figure 7. In our first attempts, we observed that our prediction models caused the decoder to include many spurious stop words in the translations. We thus restricted their source and target vocabulary to content words.

MLP-64 and perceptron do not handle large vocabulary as easily as IBM1, we thus limited their vocabulary to words appearing at least 20 times in the training corpus. The English vocabulary diminished from 133 141 to 21 915 words and the French vocabulary from 143 980 to 28 095 words. This simplification allowed us to trained our MLP in less than 3 hours on an Intel Xeon quad-core processor with a clock-rate of 2.8 GHz.

6.3 In-domain

The results of the in-domain evaluation are presented in Table 1. We first observe that all models predicting a target vocabulary get better BLEU than the baseline. Those improvements are statistically significant with a confidence of 95% according to our bootstrap resampling with replacement tests (Koehn, 2004b). We also observed that MLP-64 systems are significantly better than all the other three systems.

We compared MLP-64 translation against those of our baseline and noticed that both systems agree for one sentence out of four. The other translations usually differ in one or two words having similar senses, but MLP-64 tends to select translations that are closer to the reference translations.

[2]We limited the number of links because our training algorithm implementation had a hard time without it.

System	BLEU (%)	
	odd	pred
baseline	30.06	30.06
+ IBM1	30.32	30.65
+ PERCEPTRON	30.68	30.71
+ MLP-64	**30.86**	**31.00**

Table 1: In-domain evaluation. Bold scores are significantly better than the other scores of their column.

System	BLEU (%)	
	odd	pred
baseline	19.05	19.05
+ IBM1	20.00	**19.92**
+ PERCEPTRON	19.93	**19.82**
+ MLP-64	**20.45**	**19.89**

Table 2: Out-of-domain evaluation.

6.4 Out-of-domain

The results for news translations are presented in Table 2.

We first observe a decrease of 10 BLEU point when compared against the results of the in-domain translation. This decrease asserts the challenge we face when designing a system that should translate many genres of documents. We still observe that all models predicting a target vocabulary are better than the baseline.

The best system is MLP-64 combined with *odd* score. The translations of the system are significantly better than the translations of all the other systems according to BLEU. We observe modest gains for *pred* scores where IBM1 is the best system, but there are no significant differences among all the models predicting a target vocabulary.

6.5 Comparison to state-of-the-art

The best SMT systems of the *Fourth Workshop on Machine Translation* (Callison-Burch et al., 2009) for the English-French translation task were evaluated at 28 BLEU points comparatively to our best system which got 20.45. This is a huge difference, but news translation was the main task of this workshop and participant used much more data than we used in our experiments. It is thus not fair to compare our out-of-domains results against those ones.

We can however compare our results to those

of the *Third Workshop on Machine Translation* (Callison-Burch et al., 2008) where a news corpus was translated by systems tuned to translate parliament proceedings. The best systems for in-domain English-French translations obtained 32 BLEU points and the best systems for out-of-domain translations 20 BLEU points. Our results are thus competitive with those of state-of-the-art systems in a comparable situation.

7 Previous Works

To our knowledge, IBM1 is the first target vocabulary prediction models that was used in SMT. It was one of the best model among many others in a rescoring module for a PBSMT (Och et al., 2004).

The term *global lexical selection* was coined by Bangalore et al. (2007) who pushed the idea of target vocabulary prediction further than us. They devised a system that first predicted a set of target words and then reordered those words to produce a final translation. This system is particularly suited for languages that are not sensitive to word reordering like Hindi (Venkatapathy and Bangalore, 2009). Their system use a logistic regression model to predict the target words from the set of n-grams in the source sentence. Our $pred$ score is a softer version of this idea. It encourages the decoder to select the predicted words without forcing it and it allows the reordering to take place in the PBSMT.

Mauser et al. (2009) integrated a logistic regression model predicting target words from all the source words in a PBSMT. Using this model, they gained one BLEU point over their baseline on a Chinese to English translation task.

IBM1 is a linear regression model over probability of target words given each source words individually. Mauser et al. (2009) extended this model to condition it on source word cooccurrences (word and trigger pairs). This models improved BLEU score of one point over their baseline system. Note that our MLP consider all source words jointly, word cooccurrences are thus automatically modelled.

Patry and Langlais (2009) were the first to use a MLP to predict target words, but they did not tested it in a SMT system. They got their best results when they added a bilingual lexicon to their MLP. We tested this extension in our system but it did not improve over MLP-64.

8 Conclusion

Phrase-based statistical machine translation systems condition their scores on few source words at a time to produce their translations. While previous works used linear or logistic regression models to capture broader dependencies on the source sentence, we presented, motivated and evaluated the use of a multilayer perceptron for such a task. We opted for MLP because of their hidden units which offer an artificial representation of source sentences.

We compared three different models for target words prediction: IBM1, PERCEPTRON and MLP-64. In all our experiments, we observed a significant improvement for all these models over the baseline system, but the best of our contextual model was MLP-64 with improvements of 0.94 and 1.4 BLEU points on in-domain and out-of-domain translations respectively. These results are encouraging, especially since several choices have been made that we could revisit, thus leaving an open space for further improvements.

In this study, we controlled the size of the source and target vocabularies by selecting only words appearing at least 20 times. Since translation systems are usually good with frequent words, we would like to select our vocabularies in order to maximize the gains of the translation system.

We selected a L1 regularizer because it is known to be efficient and simple to implement. We would however like to devise another regularizer that would encourage the MLP to group together artificial representations of source sentences having similar translations.

Also, we would like to validate our model on other language pairs and other corpora and we plan to investigate the influence of the corpus size and the number of hidden units on the translation quality.

An important limitation of our MLP is their ignorance of sentence structures. One partial solution to this problem is to model source n-grams instead of source words like Venkatapathy and Bangalore (2009) suggested. This approach has its limitation because it only captures local structures and cannot consider grammatical or semantic relations. We are thus thinking about alternative ways to encode the source sentences and how these new representations could be included to our model.

References

Srinivas Bangalore, Patrick Haffner, and Stephan Kanthak. 2007. Statistical machine translation through global lexical selection and sentence reconstruction. In *ACL*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155.

Adam L. Berger, Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, John R. Gillett, John D. Lafferty, Robert L. Mercer, Harry Printz, and Luboš Ureš. 1994. The candide system for machine translation. In *HLT '94: Proceedings of the workshop on Human Language Technology*, pages 157–162. Association for Computational Linguistics.

Christopher M. Bishop. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Josh Schroeder, and Cameron Shaw Fordyce, editors. 2008. *Proceedings of the Third Workshop on Statistical Machine Translation*. Association for Computational Linguistics, June.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder, editors. 2009. *Proceedings of the Fourth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, March.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 33–40. Association for Computational Linguistics, June.

Kevin Gimpel and Noah A. Smith. 2008. Rich source-side context for statistical machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 9–17. Association for Computational Linguistics.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the European Association for Machine Translation 10th Annual Conference*, May.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180. Association for Computational Linguistics, June.

Philipp Koehn. 2004a. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*, pages 115–124.

Philipp Koehn. 2004b. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395. Association for Computational Linguistics, July.

Y. LeCun, L. Bottou, G. Orr, and K. Muller. 1998. Efficient backprop. In G. Orr and Muller K., editors, *Neural Networks: Tricks of the trade*. Springer.

Yajuan Lü, Jin Huang, and Qun Liu. 2007. Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 343–350.

Arne Mauser, Saša Hasan, and Hermann Ney. 2009. Extending statistical machine translation with discriminative and trigger-based lexicon models. In *Conference on Empirical Methods in Natural Language Processing*, August.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alexander Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir R. Radev. 2004. A smorgasbord of features for statistical machine translation. In *HLT-NAACL*, pages 161–168.

Alexandre Patry and Philippe Langlais. 2009. Prediction of words in statistical machine translation using a multilayer perceptron. In International Association for Machine Translation (IAMT), editor, *Proceedings of the Twelfth Machine Translation Summit*, aug.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1992. *Numerical recipes in C (2nd ed.): the art of scientific computing*. Cambridge University Press.

Holger Schwenk. 2007. Continuous space language models. *Comput. Speech Lang.*, 21(3):492–518.

Nicolas Stroppa, Antal van den Bosch, and Andy Way. 2007. Exploiting source similarity for smt using context-informed features. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 231–240, September.

Sriram Venkatapathy and Srinivas Bangalore. 2009. Discriminative machine translation using global lexical selection. *ACM Trans. Asian Lang. Inf. Process.*, 8(2).

David Vickrey, Luke Biewald, Marc Teyssier, and Daphne Koller. 2005. Word-sense disambiguation for machine translation. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 771–778. Association for Computational Linguistics.

R. Clint Whaley and Antoine Petitet. 2005. Minimizing development and maintenance costs in supporting persistently optimized BLAS. *Software: Practice and Experience*, 35(2):101–121, February.

Jia Xu, Yonggang Deng, Yuqing Gao, and Hermann Ney. 2007. Domain dependant statistical machine translation. In *Proceedings of MT Summit XI*, pages 515–520, September.