Phrase-based Machine Translation
Assignment 2 for 11-731

Paul Michel
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
pmichel1@cs.cmu.edu

Shruti Rijhwani
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
srijhwan@cs.cmu.edu

Abstract

We build a full statistical machine translation pipeline, which involves building a language model as well as a phrase-based translation model. Apart from the baseline implementation, we also experimented with using a Kneser-Ney smoothed language model and variations of the IBM Model 1. We achieved a BLEU score of $\mathbf{19.05}$ on the test set, a reasonable improvement over the baseline 17.98.

1 Introduction

We implemented the suggested baseline, which included a uni-directional IBM Model 1, phrase extraction and converting the phrases to a WFST.

We focused our improvements over the baseline on:

1. Better language modeling using *Kneser-Ney smoothing*.
2. Experimenting with some variations on the IBM Model 1 for word alignment.
3. Varying the maximum phrase length in the translation WFST.

The dataset for training, validation and testing is a German-English parallel corpus from the IWSLT 2016 workshop on translation. There are approximately 100000 training sentence pairs, 887 validation pairs and 1565 test sentences.

We use the validation set for tuning. For testing, we choose the model that gave the best BLEU score on the validation data. This model obtained a BLEU of 19.05 on the test set.

2 Pipeline

2.1 Language Model

For the baseline, we use the given `train-ngram.py` implementation. This uses a standard bigram language model where the probabilities are estimated using maximum likelihood estimation, that is, $p(w_2|w_1) = \frac{c(w_1,w_2)}{c(w_1)}$ (where c is the count in the training set). This model also interpolates with unigrams using a fixed interpolation weight.

As an alternative, we use Kneser-Ney smoothing (Kneser and Ney, 1995), which replaces the counts $c(w_2, w_1)$ for lower-order n-grams by the fertility $N_{1+}(\bullet w_1) = |\{w_2 : c(w_2, w_1) > 0\}|$. It also uses a discount value D to assign weights for lower-order terms. We use the interpolated Kneser-Ney model as described in Chen and Goodman (1998).

$$p(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i) - D}{c(w_{i-1})} + \alpha(w_{i-1})p(w_i)$$
$$p(w_i) = \frac{N_{1+}(\bullet w_i)}{N_{1+}(\bullet\bullet)}$$

There could be certain words that have a high frequency in the training corpus, but only exist in very specific contexts. With maximum likelihood estimation, these words would get predicted by the language model often when higher order n-grams are unknown. However, these words may only fit into certain contexts (for example, *Francisco* fits after the word *San*). Using the Knesey-ney fertility counts alleviates this problem by making the likelihood proportional to the number of *unique contexts* the unigram is seen in.

Furthermore, the Kneser Ney discount has the advantage of distributing the probability mass of higher order n-grams to the interpolation weight of lower orders across according to their count. Therefore, the interpolation weight α is different for each n-gram, rather than a fixed value like the baseline.

We also assigned unknown words a small probability mass.

2.2 IBM Model 1

We first implemented the standard unidirectional version of IBM Model 1 (from source to target language). This model did not include the ability to have null-aligned words and therefore, every target word had a source word alignment.

Based on this, we attempted using two modifications:

- Allowing words to be unaligned (null alignments). We tried this in two ways – adding a null word to the source sentence and removing low confidence alignments from the unidirectional model.
- Intersecting alignments from each direction of the Model 1.

The effects on BLEU from these variations is described in the next section.

2.3 Phrase Extraction

We used Algorithm 6 from the class notes and the pseudocode discussed to implement phrase extraction from the alignments. We varied the maximum phrase length and chose the optimal one based on validation set BLEU score.

After converting the phrases to a WFST, we used the given decoding module and did not make any modifications to that portion of the pipeline.

3 Experiments

We first implemented the baseline model as suggested and got a BLEU score of 17.91 on the validation set (Table 1), using 8 iterations for training Model 1 and 3 as the maximum phrase length. We then experimented with several modifications which are described below. The modifications that positively contributed to validation BLEU were used in the final model.

	Validation Set BLEU
Baseline	17.91
+KneserNey	18.99
+Model1Iter	19.15
+PhraseLength	**19.15**

Table 1: Validation Set BLEU Improvements with Modifications

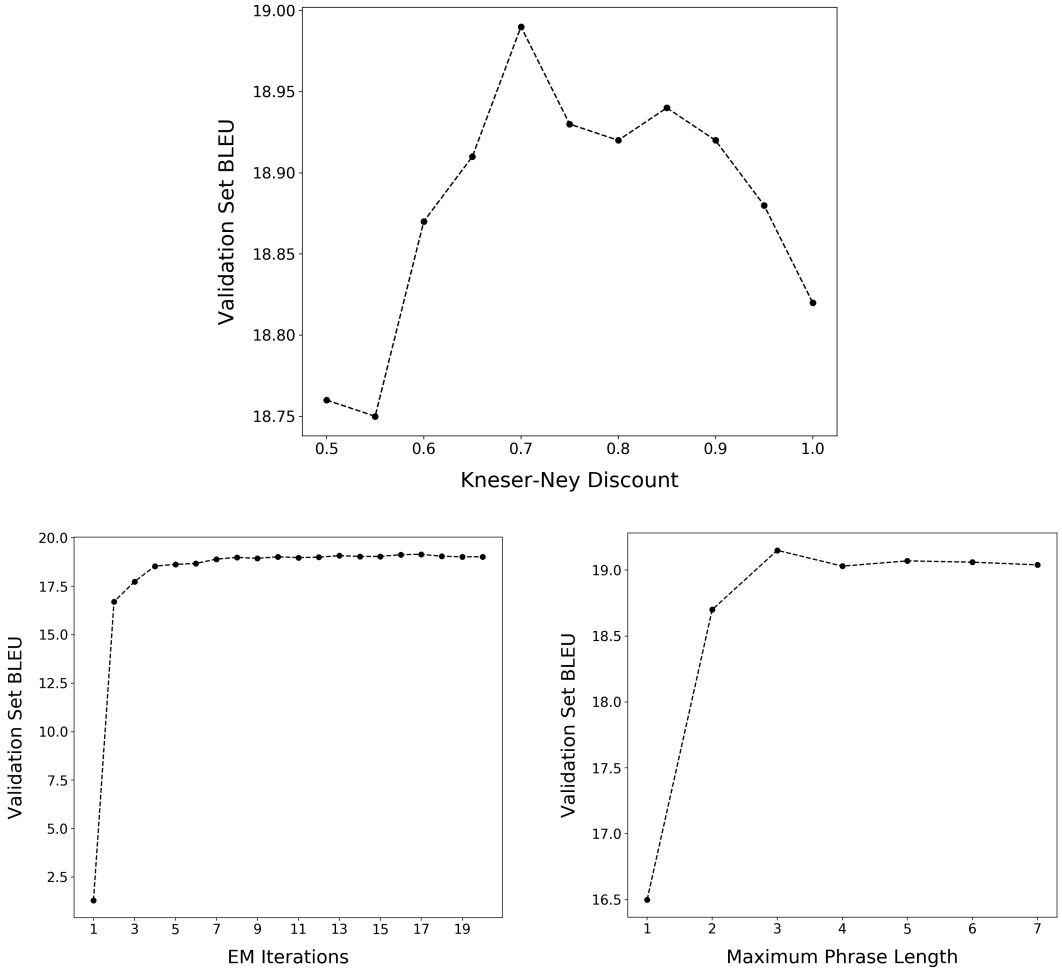

Figure 1: Effects of Modifications on Validation BLEU

3.1 Modifications

3.1.1 Kneser-Ney Smoothing

We used a Kneser-Ney smoothed bigram language model, in which the probabilities are interpolated using the equations written in the previous section. We experimented with different discount values D, between 0.5 and 1.0, and selected the best one based on the validation set BLEU. The discount chosen was 0.7, although from the figure we see that even a low discount value (0.5) gives an improvement over the baseline, as seen in Figure 1.

This modification gave us a considerable boost with the BLEU score reaching 18.99, seen in Table 1, and we use this language model in all further experiments.

3.1.2 EM Iterations in Model 1

Figure 1 shows the variation of BLEU score with the number of iterations from 1-20 used in training Model 1. We see that the BLEU score is low with a single iteration since alignments are uniformly likely then. The model quickly stabilizes after a few iterations, with slowly increasing BLEU after that. We use 17 iterations for the next experiments since the validation BLEU score obtained at this point was the highest at 19.15 (Table 1).

	Validation Set BLEU
VanillaNull	17.14
NullWeight $= 0.5$	18.76
NullWeight $= 0.2$	19.09
Confidence < 0.1	15.99
Confidence < 0.01	18.40

Table 2: Experiments with Null Alignments

3.1.3 Maximum Phrase Length

In figure 1, we see how the maximum phrase length affects BLEU score. It is interesting that the BLEU score is rather low at 16.5 when the model simply uses words (maximum phrase length $= 1$). It rises sharply even with two-word phrases to a BLEU of 18.7. It peaks when phrases are constrained to a maximum length of 3 words and is slowly decreasing after that. The best BLEU here is 19.15, like the EM iterations experiment, (Table 1) since the maximum phrase length was $= 3$ there as well.

3.1.4 Null Alignments

We added the ability to have null alignments on the target side in three ways:

1. By simply appending a `null` token to each target sentence, we could get source words aligned to null. The BLEU score obtained with this method is noted in Table 2 as `VanillaNull`. The BLEU is not as high as our previous results and on observing the phrases, we saw that there were too many phrases being generated from the null alignments, which could have contributed to noise.
2. To solve the problem described above, we tried to reduce the number of unaligned phrases by only choosing null alignments of relatively high confidence (to other target words). This was done by multiplying the probability of the null target after training the Model 1 by a fixed weight. We experimented with this `NullWeight`=0.5 and =0.2. The results are shown in Table 2, and we see that the lower the `NullWeight`, the better the BLEU score.
3. Lastly, we tried a different heuristic to get null alignments based on observations from our earlier model (unidirectional without null alignments). The probabilities of the final word alignments ranged over 0.9 to $1e^{-19}$, where over 80% had probability $>= 0.1$ and 99% had probability $>= 0.01$. Given the large range of confidence scores, replacing low confidence alignments from the original model by null alignments seemed like an interesting way to reduce noise in the phrases as well as allow unaligned words in phrases. However, as Table 2 shows, even nullifying 1% of the alignments (`Confidence` < 0.01) leads to a drop in BLEU.

From our experiments, we see that having null-aligned words, both explicitly and indirectly by pruning out low confidence alignments, results in a decrease in BLEU. It is particularly notable that to see that in all attempts, fewer unaligned words led to better BLEU. Therefore, we do not use null alignments in our final model.

3.1.5 Intersected Model 1

In order to get better alignment precision, we experimented with an intersection of IBM Model 1 alignments.

More specifically, since IBM Model 1 is asymmetric, we train two different models, one on en $\longrightarrow$ de and one on de $\longrightarrow$ en. We then retrieve the alignments and "intersect them", ie combine them by taking the "`AND`" of both alignment matrices for each sentence pair.

As it turned out, this degraded performance a lot, with a validation BLEU of 2.50 (and 3.22 on the test set).

This can be explained by the fact that intersecting the alignments leaves a lot of words aligned to null (sparse alignment matrices), which our phrase extraction algorithm didn't handle.

As a result, the extracted phrases had too high a granularity and around half of the sentences output by the model were empty, hence the low score.

This could possibly be solved by having more data.

We also briefly tried combining alignments with `OR` instead of `AND` but the results were inconclusive.

4 Results and Discussion

Based on the BLEU scores obtain on the validation data as described in the previous section, the final model we selected has:

- Kneser-Ney smoothed language modeling with a discount of 0.7
- 17 EM iterations for the Model 1
- No null alignments allowed
- Maximum phrase length as 3

This model was used to translate the test and blind data sets. The BLEU on the test set is $\mathbf{19.05}$.

4.1 Qualitative Analysis

We compare qualitatively the results on the open test set using our model and a neural model[1].

Positive	
Phrase based	i played the first set of UNK UNK .
Neural	i was playing the first sentence of bilbao .
Reference	and i played the first movement of the beethoven violin concerto .
Negative	
Phrase based	i started – so simple , to play .
Neural	so i just started to play .
Reference	so , i just started playing .

Table 3: Qualitative comparison with a neural model

Table 3 shows two examples where the phrase based model respectively over-performs (positive) and under-performs (negative) compared to the neural model.

In the positive example, the phrase based model learns to use `UNK` symbols when confronted to rare words (like "beethoven", "violin", "concerto") whereas the neural system generates a somewhat valid but inadequate translation.

On the opposite, in the negative example, the neural model translate correctly (up to the conjugation), but the phrase based model seems to use "so simple" instead of "so" (maybe an artifact of "so simple" being a high probability phrase in this context).

According to these examples, it seems that the neural model has a stronger language model in the target language, sometimes to the detriment of adequacy, whereas PBMT, although more adequate, is susceptible to "glitches" like the one in the negative example.

Of course, this comparison should be put in perspective by the fact that neither models were near the state of the art in their respective sub-fields. Other factors (time and memory cost for training and decoding) are also important in such a comparison.

[1] We used the best output from Paul's assignment 1 model

Conclusion

We trained a full-fledged phrase based translation model achieving a BLEU score of 19.05 on the test set using Kneser-Ney smoothing and hyper-parameter tuning on the validation data.

We also experimented with various ideas to improve our alignments which did not translate in a significant improvement in score.

Tasks repartition

- Shruti : Kneser-Ney, Phrase Extraction, Code for WFST, Null experiments and report
- Paul : Model 1, Model 1 variations, Phrase Extraction experiments, qualitative analysis and report

References

Stanley F Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318. Association for Computational Linguistics, 1998.

Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184. IEEE, 1995.