# Incremental Training and Intentional Over-fitting of Word Alignment

**Qin Gao**
Language Technologies Institute,
Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, PA 15213
qing@cs.cmu.edu

**Will Lewis, Chris Quirk and Mei-Yuh Hwang**
Microsoft Research
One Microsoft Way
Redmond, WA, 98052, USA
{wilewis,chrisq,mehwang}@microsoft.com

## Abstract

We investigate two problems in word alignment for machine translation. First, we compare methods for incremental word alignment to save time for large-scale machine translation systems. Various methods of using existing word alignment models trained on a larger, general corpus for incrementally aligning smaller new corpora are compared. In addition, by training separate translation tables, we eliminate the need for any re-processing of the baseline data. Experimental results are comparable or even superior to the baseline batch-mode training. Based on this success, we explore the possibility of sharpening alignment model via incremental training scheme. By first training a general word alignment model on the whole corpus and then dividing the same corpus into domain-specific partitions, followed by applying incremental training to each partition, we can improve machine translation quality as measured by BLEU.

## 1 Introduction

This paper addresses two significant problems that large-scale machine translation systems face. First, given word alignment models trained on a large amount of data, we need a method of incremental training over a small amount of new data, in order to minimize processing time associated with re-training or customizing a system. Second, we explore intentional over-fitting of word alignment models. As a special case in machine learning, word alignment for Machine Translation may actually benefit by over-fitting for a specific domain. We discuss this issue and suggest a two-step word alignment scheme to improve quality.

Word alignment is a crucial component of state-of-the-art statistical machine translation technology. Most translation models are built upon the word alignment output (Och and Ney, 2004; Chiang, 2007; Quirk and Menezes, 2006; Galley et al., 2004). Generative models (Brown et al., 1993; Vogel et al., 1996; He, 2007; Och and Ney, 2003) are widely used because of their ability to utilize sentence-aligned corpora without manual annotation. However, generative word alignment is a time-consuming process, especially in production environments where new data are constantly being added. In this case, running word alignment repeatedly for millions of sentences to gauge the impact of several thousand new sentences can be a waste of valuable resources. In this work, we explore different ways of performing fast incremental training of word alignment models and incorporating the alignment results of the new data into the existing translation models.

In our baseline machine translation system, WDHMM (He, 2007) is used for word alignment. The model is a generative HMM alignment model with a word-dependent distortion model, where parameters are estimated in a maximum likelihood fashion using the EM algorithm. The implementation is highly optimized, allowing both multi-threading and distributed computing. The basic strategy of incremental training is to utilize an existing word alignment model, updating the models on the new data. By running EM only on the smaller amount of new data, we effectively cut down the time needed for training a new system. Step-wise EM for word alignment has been explored in (Levenberg et al., 2010; Liang and Klein, 2009), where sufficient statistics on mini-batches are collected and interpolated with the general baseline. In this work, we do not store these statistics. Instead, we explore the possibility of utilizing

the model parameters directly. As we will see later in the second problem we want to address, this allows us to over-fit more radically towards incremental data or specific domains, instead of pursuing a model for the whole corpus.

There are two component models inside WDHMM: the lexical translation model $p(f_j|e_i)$, which models the probability of, say, a given French word $f_j$ and an English word $e_i$, and the distortion or transition model, which models the probability of the position in the French sentence an English word should align to given the previous English word and its position in the English sentence, denoted as $p(a_j|a_{j^*}, e_{a_{j^*}}, I)$. The training of WDHMM model is usually divided into two stages. First, IBM Model 1 is trained and used to bootstrap the WDHMM alignment, which is the second step. In the training of IBM Model 1, only the lexical translation model is used. After word alignment, an SMT system then extract parallel phrase pairs or treelet pairs. Therefore we need to answer the following questions:

1. How should we use the baseline probabilities in the incremental training scenario? Should we use the lexical translation probabilities differently from the distortion model?
2. After we get the Viterbi alignment of the training data, how can we make use of them? Should we concatenate them into the baseline system and re-extract all the phrases or can we extract a separate phrase or treelet table only from the new data and use it to augment the baseline systems' models?

While answering the questions above, another interesting thought on word alignment arises. Word alignment is a quite distinct machine learning problem because the final product is not the model but the alignment on the *training set*. Unlike other machine learning problems where the trained model is applied to new test data, word alignment methods, which are often unsupervised, only need to be tested on the training set. This characteristic means we do not need to care about the generalizability of the alignment model, and over-fitting can actually be beneficial. If we separate the corpus so that similar data falls into the same chunks, we may train a separate word alignment model on each of these chunks. Assuming that the data from the same chunk has a similar underlying distribution, we may obtain sharper distributions (i.e., over-fit the data). However, generative models require significant amount of data for reliable estimation; therefore, splitting data can lead to less accurate estimation. Given the incremental training scheme we present, it is possible to initialize or interpolate models trained on partitions of the training data with a background model trained on the whole corpus. In this case we can both produce a more accurate estimation and sharp distributions for each partition at the same time. Also, if the background model remains stable over time, we can speed up the training process by running alignment tasks for each chunk in parallel.

The paper is structured as follows: In section 2 we briefly review the HMM and WDHMM model as well as the update process. Section 3 introduces the incremental training methods, and in section 4 we discuss chunking data and training of domain-specific models. In section 5 we present the experimental results. Section 6 concludes the paper.

## 2 Background

### 2.1 HMM Model

Here we briefly review the HMM model for word alignment (Vogel et al., 1996). Given a French sentence $f_1^J = (f_1, \cdots, f_J)$ and an English tence $e_1^I = (e_1, \cdots, e_I)$, we assume a hidden alignment between the sentence pair, denoted as $a_1^J = (a_1, \cdots, a_J)$, where $a_l \in [1, I]$. The alignment link $a_j$ means the French word $f_j$ aligns to English word $e_{a_j}$. The translation process is modeled as a noisy channel model, observing the French sentence and the alignment given the English sentence:

$$P(f_1^J|e_1^I) = \sum_{\forall a_1^J} P(a_1^J, f_1^J|e_1^I) \quad (1)$$

The HMM model is a parametric form of equation (1). The probability $P(a_1^J, f_1^J|e_1^I)$ is split into two parts. The first is a lexical translation el $p(f_j|e_i)$, which only depends on the French and English word. The second is the distortion or distortion probability $p(a_j - a_{j-1}|a_{j-1}, I)$, which gives a distribution over the distance between the

English words that the current and previous French words aligned with and the length of the source sentence. Each pair $\left(a_j, e_{a_j}\right)$is considered to be an HMM state that outputs the French word $f_j$. Given the assumptions, formula (1) can be expanded as:

$$\begin{aligned} P(f_1^J|e_1^I) &= \sum_{\forall a_1^J} P(a_1^J f_1^J|e_1^I) \\ &= \sum_{\forall a_1^J} \prod_{j=1}^{J} \left[p(l|a_{j-1}, I) p\left(f_j \middle| e_{a_j}\right)\right] \end{aligned} \tag{2}$$

where $l = a_j - a_{j-1}$. In addition, we can allow the French word to be aligned with a virtual "null" word, which introduce a new probability $p_0$ for jumping to that state.

The parameter set of the HMM model is: $\Theta = \{p(l|a_{j-1}, I), p(f_j|e_i)\}$ can be estimated by MLE:

$$\Theta_{ML} = \arg\max_{\Theta} p(F|E, \Theta) \tag{3}$$

where F, E are sets of parallel French and English sentences. The estimation can be done efficiently using EM algorithm (Rabiner, 1989). Basically, in the E-step, we calculate the posterior probability of each "event" for each sentence. For example, we calculate and sum the probability of

$$c(f'|e') = \sum_{\left(e_1^I, f_1^J\right) \in (E,F)} \Pr(f_j = f', e_{a_j} = e'|f_1^J, e_1^I, \Theta) \tag{4}$$

And then normalize the counts to update to a new model.

$$p^*(f'|e') = \frac{c(f'|e')}{\sum_{\forall e} c(f'|e)} \tag{5}$$

And for distortion model:

$$c(l) = \sum_{\left(e_1^I, f_1^J\right) \in (E,F)} \sum_{j=1}^{J-1} \sum_{1}^{I} \Pr(a_j = i, a_{j+1} = i + l | f_1^J, e_1^I, \Theta) \tag{6}$$

$$p^*(f'|e') = \frac{c(l)}{\sum_l c(l)} \tag{7}$$

The Forward-Backward algorithm introduced in (Rabiner, 1989) can be used to estimate the posterior probability in equation (6).

### 2.2 WDHMM

WDHMM is a natural extension to the HMM alignment model where the distortion model depends on the previously generated lexical item. That is, the first probability entry in the right hand side of equation (2) becomes: $p\left(l\middle|e_{a_{j-1}}, a_{j-1}, I\right)$. However, in this case, the parameter estimation can become prone to data scarcity, since we have much more distortion parameter than HMM especially for rare words. In that case, the number of samples to estimate $p\left(l\middle|e_{a_{j-1}}, a_{j-1}, I\right)$ is too limited to allow a statistically significant estimation. In (He, 2007), a Maximum a-Posteriori training method (Gauvain and Lee, 1994) is introduced, which estimates the probability as follows:

$$p_{MAP}(l|a_{j-1}, e, I) = \frac{c(l; e) + v_{a_{j-1},l} - 1}{\sum_l c(l; e) + \sum_l v_{a_{j-1},l} - I} \tag{8}$$

where $v_{a_{j-1},l}$ is the hyper-parameter for a Dirichlet prior over the multinomial distribution. The parameter is set using the word-independent distortion probability:

$$v_{a_{j-1},l} = p(l|a_{j-1}, I) \tag{9}$$

In the current implementation, the prior is set to a word-class dependent probability:

$$v_{a_{j-1},l} = p\left(l\middle|a_{j-1}, w\left(e_{a_{j-1}}\right), I\right) \tag{10}$$

Therefore, the final update function is as follows:

$$\begin{aligned} &p_{MAP}(l|a_{j-1}, e, I) \\ &= \frac{c(l; e) + \tau \cdot p\left(l\middle|a_{j-1}, w\left(e_{a_{j-1}}\right), I\right)}{\sum_l c(l; e) + \tau} \end{aligned} \tag{11}$$

Our implementation of WDHMM is highly optimized to allow multi-threading as well as distributed computing using MPI. In the MPI setup, a single head process first reads the corpus and splits it into several chunks which are sent to all the participant leaf processes. Leaf processes send counts back to the head node, and the head node performs normalization before sending back the new parameters. The training is also divided into two stages: first Model 1 is trained and stored, and then WDHMM loads this model and performs training of the WDHMM model.

## 3 Incremental training

In a production environment, we frequently encounter the situation that a relatively small amount of new data is added to a larger, rather static corpus. Conventionally, the word alignment process will be re-run over the concatenated corpus. This setup is sub-optimal because we do not take ad-

vantage of existing model. Instead we go through the time-consuming process of calculating the posterior probabilities over the whole corpus. We may save significant computational resources if we can just run word alignment on the small new corpus. However, as a generative model, there is no data like more data: training models solely on the new corpus will result in a poorly trained model that degrades the performance of the system. Another extreme is to simply use the existing model, aligning the new corpus without parameter re-estimation. This is also problematic because it does not reflect the different distribution the new corpus might have, especially the new vocabulary entries in the new corpus. Therefore, a better solution could be to both leverage the existing models (herein referred to as Background Models) in some way and also run EM only on the small corpus.

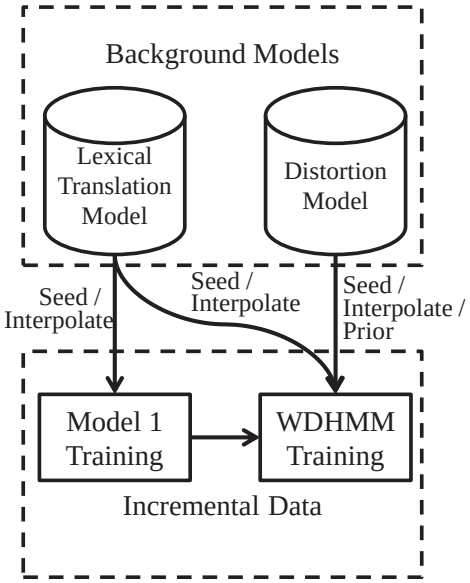


Figure 1: Possible ways of utilizing background models in incremental training process.

### 3.1 Using background models: Seeding vs. Interpolation vs. Distortion Prior

Given the two components of the WDHMM model and the two training steps, there are multiple ways to use a background model, as demonstrated in Figure 1. First, for the lexical translation model, we can simply just use the background model to initialize the alignment parameters. Before the first iteration of training, the probability $p(f_j|e_i)$ will be initialized as $p(f_j|e_i) = p_{\text{bg}}(f_j|e_i)$, where the subscript $bg$ refers to "background". There are several exceptions we need to deal with. If $e_i$ is not in the background vocabulary, then corresponding entries will have a uniform probability. If $e_i$ is in the background vocabulary but $f_j$ is not, then $p(f_j|e_i)$ is given a small floored value. After initialization, the training and updating process will continue as usual. Seeding has the advantage of breaking ties as mentioned in Section 1 and speeding up the convergence even with Model 1.

Alternatively we can train $p_{\text{new}}(f_j|e_i)$ via interpolation with a frozen $p_{bg}(f_j|e_i)$ defining the posterior probability as:

$$p(f_j|e_i) = \lambda \cdot p_{bg}(f_j|e_i) + (1-\lambda) \cdot p_{new}(f_j|e_i) \qquad (12)$$

during alignment and updating only $p_{new}(f_j|e_i)$ during each training iteration. The constant $\lambda$ can be chosen empirically.

Finally we can perform both seeding and interpolation. Also, we can either choose to perform these operations starting at Model 1, or to bypass the IBM Model 1 training and directly do interpolation/bootstrapping from WDHMM training stage.

For the distortion probability, we can also use the two methods mentioned above. In addition, consider formula (11), where we use the word class dependent probability as a prior. Instead of interpolating the background model, we can use an "interpolated prior", which involves another hyperparameter $\tau_2$, so that:

$$\begin{aligned} p_{\text{MAP}}(l|a_{j-1}, e, I) &= \frac{c(l;e) + \tau \cdot p_{\text{int}}(l, a_{j-1}, e)}{\sum_l c(l;e) + \tau} \\ p_{\text{int}}(l|a_{j-1}, e) &= \tau_2 \cdot p_{\text{bg}}(l|a_{j-1}, e, I) + (1-\tau_2) \\ &\quad \cdot p\left(l\middle|a_{j-1}, w\left(e_{a_{j-1}}\right), I\right) \end{aligned} \qquad (13)$$

### 3.2 Using Viterbi alignments: One vs. Multiple Translation Tables

After identifying the Viterbi alignments of the new corpus, there are also different ways of utilizing them. The most straightforward way is to concatenate them with the Viterbi alignments of the background corpus. This method can be considered a test of one of the different ways of utilizing a background model. Our target is to speed up the process without degrading the quality of translation. Therefore the concatenated Viterbi alignment should perform close to that of when we concate-

nate the new corpus with the background data and run global word alignment on it. However, the method can still be time-consuming because the phrase or treelet extraction can take a long time. Therefore, an alternative method is to extract a phrase table or treelet table separately for the new corpus, and use it in both tuning and decoding with two separate feature sets. The ideal scenario is that, when new data arrives, we continually train a new but relatively small phrase or treelet table on the recently updated portion of the data and include this model into the system. When the size of new corpus achieves a certain size, we combine the new with the old data, and retrain a new background model.

In Section 5, we will show the performance comparisons of the methods mention above. It is worthy to mention beforehand that we observed in some cases the performance of incremental training are better than batch-mode training. This observation leads to the proposal of training domain specific over-fit models that may improve the quality of word alignment.

## 4 Intentional Over-fitting

As a machine learning problem, word alignment for machine translation is quite special given its applications. Usually, in machine learning problems, our goal is to learn a generalizable model that is evaluated on a new test data. However, in unsupervised word alignment, we instead hope to optimize the quality of the alignment on the training set. We have no need to apply these alignment models to unseen test data in most cases. Therefore, the over-fitting problem, which is a significant detriment to many other machine learning problems, can actually be beneficial to alignment quality.

It is worth asking whether statement "there's no data like more data" still holds. Or in other words, if we have two corpora from radically different sources with very different distributions, should we put them together and train a "generalizable" word alignment model, which may have a flatter distribution on both sources, or should we train them separately? The question actually presents a dilemma because we must either train models on homogeneous corpora with less data, or train a model on a heterogeneous corpus with more data. However, both extremes may be sub-optimal. In this section we aim to derive a training scheme that takes advantage of more data yet adapts to the distribution of a specific domain.

With the incremental training methods we discussed above, it is possible to replace the "background model" with a model trained on the whole corpus and perform "incremental" training on each of the separate domains. By doing so, the knowledge we gain from the entire dataset can help the alignment of each of the separate domains; yet domain-specific information may become more prominent within each domain-specific training. The method maintains a balance between general and domain-specific training. Furthermore, since the general model can be stable, it can also benefit training speed when only one of the domains has changed. In that case we can just re-align the specific domain instead of all of them.

In practice, we first train the general model with all the data, and then based on some criteria, split the data into partitions. In this paper, we manually partition the corpus based on the source of the data. More sophisticated methods can be used to classify the data; however, that is beyond the scope of this paper. We align the data of each domain by using one of the incremental training methods discussed. After that, the Viterbi alignments are concatenated and the phrase or treelet extraction is done as usual. The pipeline is demonstrated in Figure 2.

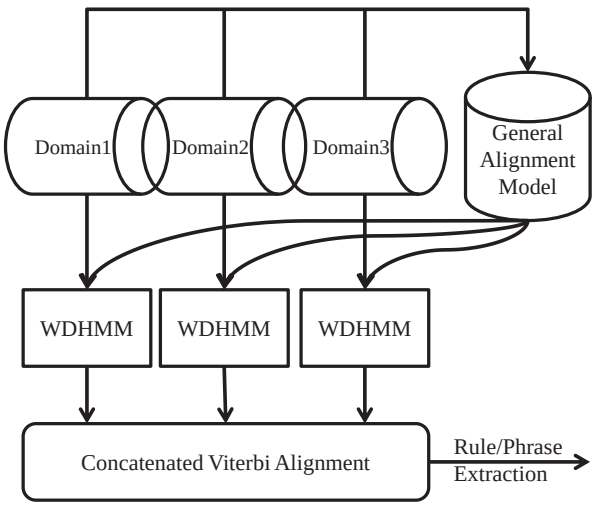


Figure 2: Pipeline of the intentional over-fitting for word alignment. The general model is trained on all data and then adapted to each domain. The final alignment results are concatenated and ready for further process such as rule extraction.

## 5 Experiments

We design the experiments to examine the following hypotheses:

1. By applying incremental training on small amount of incremental data, we can get **comparable** word alignment and translation quality comparing to batch-mode training, as described in section 3.1.
2. By using two separate translation tables, extracted from the baseline data and the incremental data, we can get **comparable** performance compared to concatenating the Viterbi alignment output and generating a single translation table.
3. By over-fitting word alignment, we can **improve the performance** of translation system compared to the baseline.

### 5.1 Experiments on incremental training

| | *Test1* | *Test2* |
|---|---|---|
| System 1: Background+tech batch | 59.38 | 59.72 |
| System 2: Background only | 54.18 | 54.57 |
| System 3: Independently-aligned | 58.62 | 58.94 |

Table 1: BLEU scores of three baseline systems on English-Indonesian translation. System 2 provides the background alignment model for future incremental training in Table 2.

In this experiment we use an English-Indonesian machine translation task with moderate data size to allow numerous experiments on different schemes. We split the data into two parts: the background, which has 1.65M sentence and the tech data with 251K sentences. The development set contains 500 sentences, and the two test sets contain 1000 and 950 sentences, respectively. All the development and test sets are from general domain. We use the dependency treelet system described in (Quirk and Menezes, 2006) to perform the translation task. For comparison, we present three contrastive systems:

1. The system trained with both background and tech data with batch-mode training;
2. The system trained with only the background data;
3. The system in which the tech data are aligned independently of the background data, i.e. the tech data are separately aligned without using background model or data. The Viterbi output of the tech data is concatenated with that from System 2 to continue rule extraction.

From Table 1 we make the following observations: First, the test set is sensitive to the training data from tech domain. Excluding the tech domain data from training (System 2) causes significant drop in performance. Secondly, aligning background data and tech data separately causes a drop in the quality of translation. The only difference between System 1 and System 3 is in the word alignment. To prove our hypotheses 1, the incremental training should perform better than System 3, and approach System 1.

| Start From IBM Model 1 Training | | | |
|---|---|---|---|
| *Lexical model* | *Distortion model* | *Test1* | *Test2* |
| Seeding | Ignored | 59.37 | 59.61 |
| Interpolate | Ignored | **59.57** | **59.87** |
| **Interpolate** | **Seeding** | **59.69** | **60.01** |
| Interpolate | Prior | 59.37 | 59.68 |
| Interpolate | Interpolate | 58.95 | 59.22 |
| Start Directly From WDHMM Training | | | |
| Interpolate | Interpolate | 58.72 | 58.98 |
| Interpolate | Seeding | 58.84 | 59.12 |
| Interpolate | Prior | 58.92 | 59.16 |

Table 2: Most informative results for incrementally training on the English-Indonesian tech parallel data, with System 2 as the background model.

We compare various incremental training schemes on the tech training data. All of them use the word alignment model from System 2 as the "background model". Similar to System 3, the Viterbi output is concatenated and rules are extracted from the total set. In other words, we present only one translation table per system in this set of experiments. Since there are a large number of possible combinations of the training schemes, here we report only a subset of informative results in Table 2. As we can see from the results, various ways of smoothing the lexical translation model and the distortion model all improve over System 3. The best configuration is to start from Model 1 training, with the lexical model interpolated with that from background and seeding the distortion model at the same time. To our delight, it is even better than the baseline System 1. This inspired us to propose the over-fitting method mentioned in Section 4.

### 5.2 Experiments on separate rule tables

In this experiment we use the English-Norwegian translation task with larger data. Similarly the corpus is split into background data with 2.03M sentence pairs and 46.3M words, and tech domain data with 1.89M sentence pairs and 46.0M words. We also ran experiments with the three contrastive systems and the best incremental scheme we got from previous experiment. Then, to validate the hypothesis of separate rule tables, we extract a second dependency treelet table. Both dependency treelet tables are use in tuning and decoding, with separate sets of features. The development set has 1000 sentence pairs. The two test sets has 2500 sentence pairs and 2300 sentence pairs, all of them are from general domain. The experimental results are shown in Table 3.

| *Usage of Lexical Model* | *Usage of Transition Model* | *Test Set 1* | *Test Set 2* |
|---|---|---|---|
| Contrastive Systems | | | |
| System 1: Background+tech batch | | 56.50 | 55.72 |
| System 2: Background only | | 54.01 | 53.41 |
| System 3: Independently-aligned | | 55.49 | 55.49 |
| Start From IBM Model 1 Training | | | |
| Interpolate | Initialize | 56.37 | 55.57 |
| Two Treelet Tables | | | |
| **Interpolate** | **Initialize** | **56.59** | **55.86** |

Table 3: Experimental results of separate rule tables on the English-Norwegian translation task.

From the results shown in Table 3, using separate treelet tables yields similar performance compared to concatenating the Viterbi alignments. By combining incremental word alignment training and separate rule tables, we effectively limit the entire training pipeline to only the new data set. That is, training can occur without access to the background data. Depending on the amount of background data, this improved pipeline can save a large amount of processing time and resource while producing similar or even superior results.

### 5.3 Experiments on Intentional Over-fitting

We perform experiments on intentional over-fitting on English-Norwegian, English-Arabic, Arabic-English, English-Chinese, English-Ukrainian and English-Vietnamese translation tasks. The statistics of the corpus are listed in Table 4.

| *System* | *Sent (M)* | *Word (M)* | *Dev Set* | *Test 1* | *Test 2* | *Test 3* | *No. Domains* |
|---|---|---|---|---|---|---|---|
| EN-AR | 11.1 | 473 | 2500 | 5000 | 4882 | 999 | 6 |
| AR-EN | 11.1 | 473 | 2500 | 5000 | 5000 | 999 | 6 |
| EN-NO | 2.03 | 46.3 | 1000 | 2500 | 2300 | N/A | 5 |
| EN-CH | 12.9 | 370 | 2000 | 5000 | 972 | 1000 | 12 |
| EN-VI | 3.48 | 68.0 | 2000 | 1000 | 508 | 995 | 4 |
| EN-UK | 2.62 | 56.6 | 500 | 1000 | 837 | 999 | 5 |

Table 4: Statistics of corpus used in the intentional over-fitting experiments.

The Arabic-English system is phrase-based, while the other systems use dependency-treelet system. As mentioned before, in all of the systems, all the data are used in training the background model and then apply the incremental training scheme (Interpolating background lexical translation model and initialize transition model with background model. The experimental results are listed in Table 5.

| *Lang. Pair* | *System* | *Test Set 1* | *Test Set 2* | *Test Set 3* |
|---|---|---|---|---|
| EN-AR | Baseline | 15.24 | 15.12 | 22.86 |
| | Over-fit | 15.40 ↑ | 15.27 ↑ | 23.03 ↑ |
| AR-EN | Baseline | 28.14 | 27.94 | 27.38 |
| | Over-fit | 28.46 ↑ | 28.23 ↑ | 27.99 ↑ |
| EN-NO | Baseline | 56.50 | 55.72 | N/A |
| | Over-fit | 56.95 ↑ | 56.18 ↑ | N/A |
| EN-VI | Baseline | 50.44 | 56.75 | 32.13 |
| | Over-fit | 51.19 ↑ | 57.60 ↑ | 32.33 ↑ |
| EN-CH | Baseline | 18.86 | 12.22 | 13.56 |
| | Over-fit | 18.77 ↓ | 12.33 ↑ | 13.51 ↓ |
| EN-UK | Baseline | 37.59 | 37.49 | 13.64 |
| | Over-fit | 37.41 ↓ | 37.17 ↓ | 13.42 ↓ |

Table 5: Experiment results of intentional over-fitting on English-Arabic translation task.

From the results we observe improvements on both large-scale corpus and moderate size data. The largest improvement we get is on English to Vietnamese. However the method does not work well for English-Chinese and English-Ukrainian. Consider the "domain" in this paper is defined primitively by the source (or as simple as the file name) of corpus, the method still has potential to improve by optimizing the domain clustering, to ensure similar corpus goes into the same chunk. Again, domain clustering is beyond the scope of this paper.

Although the current setup requires a two-stage word alignment that increases the resource consumption, the first step does not need to be performed frequently with a stable background model.

## 6 Conclusion

In this paper we discussed two problems of word alignment. First of all, we investigate the possibility of incrementally training word alignment models on small updated corpora. Our experiments suggest that applying proper initialization and interpolation of models may alleviate the need for running EM over the whole corpus while achieving comparable results. Building two separate treelet tables with independently estimated feature values can further reduce the processing time. By combining the two methods we avoid the necessity of accessing background data during incremental training.

Secondly, we discussed intentionally over-fitting word alignment. Given the distinct property that word alignment for machine translation has, we encourage over-fitting of models, which other Machine Learning problems strive to avoid. Such over-fitting can actually improve the quality of word alignment. By training a general model on all the data and applying one of the incremental training schemes on each domain, we observed improvements on BLEU scores on four of six different machine translation tasks.

The readers may be interested in the hyper-parameters of the incremental training. However, given varying quality of background model and amount of incremental data, it is hard to pick a single universal weight. It would be an interesting future research direction on how to automatically and dynamically tune the weights according to the size of background and incremental data.

Also, in this paper, we use a relatively simple and straightforward method to split the data into domains. Another promising research direction is to automatically cluster the corpus by word distribution to further exploit intentional over-fitting.

## References


Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra and Robert L. Mercer, 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2), pp.263-312.

David Chiang, 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2), p.201–228.

Michel Galley, Mark Hopkins, Kevin Knight and Daniel Marcu, 2004. What's in a translation rule? In *HLT-NAACL 2004: Main Proceedings*., 2004.

Jean-Luc Gauvain and Chin-Hui Lee, 1994. Maximum a Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains. *IEEE Transations on Speech and Audio Processing*, 2(2), pp.291-300.

Xiaodong He, 2007. Using Word-Dependent Transition Models in HMM based Word Alignment for Statistical Machine Translation. In *Proceedings of 2nd ACL SMT Workshop*., 2007.

Abby Levenberg, Chris Callison-Burch and Miles Osborne, 2010. Stream-based translation models for statistical machine translation. In *Proceedings of HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Los Angeles, California, 2010. Association for Computational Linguistics.

Percy Liang and Dan Klein, 2009. Online EM for unsupervised models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Boulder, Colorado, 2009. Association for Computational Linguistics.

Franz J. Och and Hermann Ney, 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1), pp.19-51.

Franz J. Och and Hermann Ney, 2004. The Alignment template approach to statistical machine translation. *Computational Linguistics*, 30, pp.417-49.

Chris Quirk and Arul Menezes, 2006. Dependency Treelet Translation: The convergence of statistical and example-based machine translation? *Machine Translation*, 20, p.43–65.

Lawrence Rabiner, 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, pp.257-86.

Stephan Vogel, Hermann Ney and Christoph Tillmann, 1996. HMM-Based Word Alignment in Statistical Translation. In *Proceedings of International Conference on Computational Linguistics (COLING)*., 1996.