MT Project Proposal: Using Neural Networks to Learn Word Alignments

Bailey Parker
Johns Hopkins University
bailey@jhu.edu

Vivian Tsai
Johns Hopkins University
viv@jhu.edu

William Watson
Johns Hopkins University
wwatso13@jhu.edu

Abstract

We seek to explore the word alignment problem with the help of neural networks. More specifically, we want to see if the original Expectation-Maximization (EM) approach to word alignment can be transcribed as a neural network in a supervised and unsupervised setting.

1 Introduction

Word Alignment seeks to match individual words in a parallel corpus such that the Alignment Error Rate (AER) is minimized. In a previous assignment, our task was to implement IBM Model 1, based on the Expectation-Maximization (EM) algorithm. Following IBM Model 1, we implemented a reparametrization of IBM Model 2 that favors alignments along the diagonal. Finally, we implemented an Alignment by Agreement Model that trained a French to English and English to French model and combined the results to make better alignments.

Our proposal for this project is to replace the EM algorithm and reparametrization of IBM Model 2 with a neural network architecture. In addition, the model will incorporate Alignment by Agreement. We will use PyTorch to build the model and loss functions. The key concept is that we can create a loss function that the network will learn the optimal parameters to learn word alignments from a corpus of parallel text.

2 Background

Our approach is inspired by current research in topic modeling with Latent Dirichlet Allocation (LDA) models. LDA models are generative probabilistic models that model topics across distributions of words in a corpus. LDAs are based on variational methods and EM for Bayes parameter estimation.

applying a NN to approximate the posterior distributions.
see apibm - apply tech to lda.
[1] [3]

Current research attempts to supplant the probabilistic modeling with a neural network architecture.

Additional inspiration comes from Variational Auto-Encoders (VAE).

3 Original Formulation

IBM Model 1 estimates the translation probabilities of the data through the EM algorithm[1]. The EM algorithm is useful in discovering and learning the parameters of a latent variable model. In the original IBM Model 1, this is the translation probability distribution $t(e_i|f_j)$.

The work of Dyer et al. [2] added an effective reparameterization of IBM Model 2 to give an alignment distribution a. We define for a French sentence $\mathbf{f}$ of length $n = |\mathbf{f}|$, an English sentence $\mathbf{e}$ of length $m = |\mathbf{e}|$, with precision parameter λ, the alignment distribution a as follows:

$$h(i, j, m, n) = -\left|\frac{i}{m} - \frac{j}{n}\right| \qquad a(i, j, m, n) = e^{\lambda h(i,j,m,n)} \tag{1}$$

[1] http://mt-class.org/jhu/assets/papers/alopez-model1-tutorial.pdf

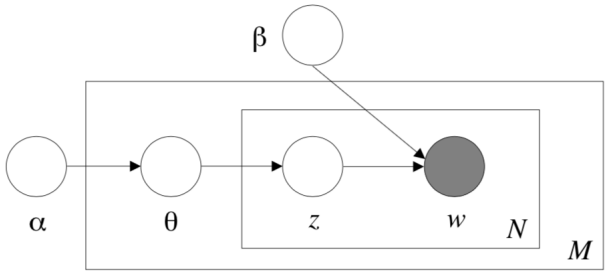

Figure 1: Graphical model representation of LDA from [1], for topic mixture θ (for a collection of M documents); set z of N topics; set w of N words, and corpus-level parameters α and β. The boxes are "plates" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.

This formulation gives us a position aware distribution that favors alignments towards the diagonal for a given English word e_i and French word f_j. The precision parameter λ controls how strongly the model prefers the diagonal.

Once we are done training the model, to make an alignment guess, we must find the alignment that returns the maximum probability. For some alignment a_i:

$$\hat{a}_i = \arg\max_{a_i} t(e_{a_i}|f_j) \cdot a(i, j, m, n) \tag{2}$$

where $t(e_{a_i}|f_j)$ is the translation probability, and an alignment distribution a with parameters λ and p_0. This will give us the English word e_{a_i} that is the most likely translation of the French word f_j.

Finally, incorporating the Alignment by Agreement model improvement was inspired by the work done by Liang et al. [4]. However, our original implementation was strongly influenced by the pseudocode used in the `Grow-Diag-Final` Alignment Heuristic by the Moses Statistical Translation System[2].

We trained an EM model in both directions, one for English to French, $A_{E\to F}$, and another for French to English, $A_{F\to E}$. Using the alignments generated by each individual model, we then proceed to take the intersection of each model's alignments.

$$Intersection = A_{E\to F} \bigcap A_{F\to E} \tag{3}$$

$$Union = A_{E\to F} \bigcup A_{F\to E} \tag{4}$$

This effectively gives us an alignment matrix that both models favor strongly, i.e. agree on. From this intersection matrix, we can then use the `GROW-DIAG-FINAL` algorithm to fill in the remaining gaps from the union of alignments.

[2] `http://www.statmt.org/moses/?n=FactoredTraining.AlignWords`

4 Our Formulation

4.1 Inspiration

We can formulate the word alignment problem as a probabilistic model, similar to LDAs and VAEs.

We want to use neural networks to encode words, create complex relations and functions to learn the parameters of the underlying distirbution to maximize some likelihood function. We intend to use PyTorch to implement the embedding layers, alignment distribution, and translation weights. In addition, we will encdoe this likelihood function as a loss function that the network will optimze its parameters for, with respect to the distirbution, formulation, and likelihood function.

4.2 Alignment Distribution

We can describe our alignment distribution as a function of target position i, source position j, and learnable parameters λ and p_0. Here we define λ as the alignment distortion parameter, and p_0 and the probability of aligning to the null token. From Dyer's paper, parameter values were selected as $\lambda = 4$ and $p_0 = 0.08$ for the entire corpus, unlike our formulation which would allow the parameters to be updated via backpropagation. As a result, the alignment distribution a can be formulated as such:

$$a(i, j \mid \lambda, p_0) = \begin{cases} p_0 & \text{if } \textit{null} \\ (1 - p_0) \cdot e^{-\lambda |\frac{i}{m} - \frac{j}{n}|} & \text{else} \end{cases} \tag{5}$$

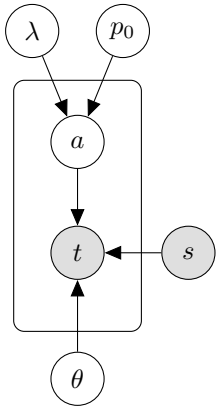

Figure 2: Our Probabilistic Plate Diagram for Word Alignment, where t is target, s is source, θ is the translation model, λ is a position distortion parameter, p_0 is the null probability, and a is the alignment distribution.

4.3 Neural Network Architecture

4.3.1 Word Embedding

4.3.2 Maximium Likelihood Function

USEFUL EQUATIONS FOR FUTURE REFERENCE:
Softmax:

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{6}$$

$$\sigma_i(\mathbf{x}) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{7}$$

Log Softmax:

$$\log\left(Softmax(x_i)\right) = x_i - \log\left(\sum_j \exp(x_j)\right) \tag{8}$$

$$\log \sigma_i(\mathbf{x}) = x_i - \log\left(\sum_j \exp(x_j)\right) \tag{9}$$

For discrete probability distributions P and Q defined on the same probability space, the Kullback-Leibler (KL) divergence from Q to P is defined to be:

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i) \log\left(\frac{P(i)}{Q(i)}\right) \tag{10}$$

In other words, it is the expectation of the logarithmic difference between the probabilities P and Q, where the expectation is taken using the probabilities P.

5 Data

6 Training

6.1 Supervised Training

6.2 Unsupervised Training

7 Expectation

$$\begin{aligned} \textit{le chat} &\mapsto \textit{the cat} \\ \textit{est} &\mapsto \textit{is} \\ \textit{noir} &\mapsto \textit{black} \end{aligned} \tag{11}$$

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[2] C. Dyer, V. Chahuneau, and N. A. Smith. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, 2013.

[3] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[4] P. Liang, B. Taskar, and D. Klein. Alignment by agreement. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 104–111. Association for Computational Linguistics, 2006.