Natural Language Processing 2 Project 1

Arthur Bražinskas
University of Amsterdam
arthuras.brazinskas@student.uva.nl

Bryan Eikema
University of Amsterdam
bryan.eikema@student.uva.nl

Abstract

In this research we compare the IBM 1 and 2 models as word-by-word alignment models. We experiment with the classical versions of these models and some modifications, such as adding a Dirichlet prior over the categorical translation parameters. We show that the IBM 2 models outperform IBM 1 models in terms of Alignment Error Rate on a held-out test dataset, and that the IBM 2 model is sensitive to initialization. We then continue to empirically show that adding a prior on the lexical translation parameters can be beneficial for the performance of the latter model.

1 Introduction

The IBM models were originally introduced as word-by-word statistical translation models [Brown et al., 1993]. They introduced the concept of word-by-word alignments between pairs of sentences in a parallel corpus. Nowadays, the IBM models are not used for translation anymore, but the alignments they produce can still be of value as inputs to other algorithms.

The document is structured as follows. We start by a brief introduction of IBM 1 and 2 models both with and without a prior, we assume that the reading is familiar with EM [Dempster et al., 1977] and variational inference involving the *mean-field* assumption. Afterwards, we present our experimental setup and the results. Finally, we finish up with a discussion and the conclusion.

2 Model

2.1 IBM 1

The IBM 1 model is the first IBM model and the simplest one. It makes large independence assumptions that make the likelihood surface have only global optima. The IBM 1 model can be seen as a constrained mixture model. The mixture components are the English words, but are constrained to the m English words that appear in the English sentence. The alignments a_j are only dependent on the the sentence length of the English and the French sentence, and act as indicators of the mixture component that generated French word f_j, essentially as mixture weights. The IBM 1 model also adds a special `NULL` token as the 0_{th} mixture component in the English sentence. The `NULL` token generates French words that do translate into English. An example would be the French negation "ne ... pas", which would typically only translate into a single word, e.g. "not", in English. The graphical model corresponding to this constrained mixture model is shown in Figure 1, in which S denotes the entire parallel corpus.

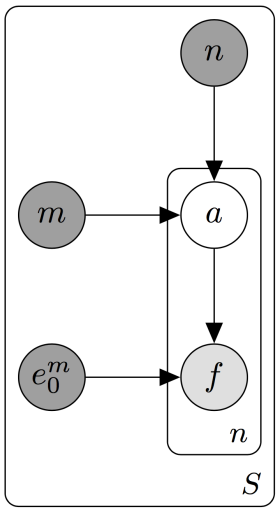


Figure 1: Graphical model for IBM 1

Note the independence assumptions made in the IBM 1 model. The alignments are independent of each other, and do not depend on the English or

French words, but only on the sentence lengths. Furthermore, the French words are independent of each other, and are only dependent on the English words that generated them. The sentence pairs in the parallel corpus are assumed to be i.i.d. The incomplete-data likelihood of the model can be written in a simple form, shown in Eq. 1 for a single sentence pair.

$$P(f_1^n|e_0^m) = \prod_{j=1}^{n} \sum_{a_j=0}^{m} P(a_j|m,n)P(f_j|e_{a_j}) \quad (1)$$

The lexical translation probabilities are distributed with categorical distributions for each English word, $P(F \mid E = e) = Cat(F \mid \theta_e)$. The alignment distribution in IBM model 1 is assumed to be distributed uniformly, i.e. all possible alignments are equally likely. This is a strong simplifying assumption, but causes the likelihood to have only global optima.

We train the IBM 1 model using the EM algorithm. The EM algorithm for a categorical distribution takes the form of the normalized expected co-occurrences of words e and f with respect to the posterior distribution over the latent alignments. We can do this, because the posterior distribution over the alignments takes a simple form, as shown in Eq. 2. The update equation for a categorical distribution parameter $\theta_{f \mid e}$ is shown in Eq. 3.

$$P(a_j|m,n) = \frac{P(a_j \mid m,n)P(f_j \mid e_{a_j})}{\sum_{i=0}^{m} P(a_j = i \mid m,n)P(f_j \mid e_{a_j})} \quad (2)$$

$$\theta_{f \mid e} = \frac{\mathbb{E}[n(e \to f) \mid A_1^n]}{\sum_{f'} \mathbb{E}[n(e \to f') \mid A_1^n]} \quad (3)$$

Effectively, this means that in the E-step we're counting how often two words e and f co-occur in the parallel corpus, weighted by how likely it is that those words are aligned using the current parameter values $\theta_{f \mid e}$. The posterior probability is really only influenced by the current lexical translation probabilities $\theta_{f \mid e}$, because the alignment probabilities are assumed to be uniform. In the M-step, we update parameters $\theta_{f \mid e}$ by normalizing the expected counts of e and f co-occurring, by dividing by the expected number of times that e co-occurs with any word f in the corpus.

2.2 IBM 1 Variational Bayes

One alternative to the classical IBM 1 model is to introduce a prior over categorical parameter θ that corresponds to the translation model. This approach has several benefits such as an ability to encode prior knowledge and prevention of overfitting [Bishop, 2006]. We can encode our prior knowledge by choosing specific hyper-parameters of a prior, in our case the prior is Dirichlet, which is parametrized by a vector α, and if its values are choose to be small, it will encode an idea that words are more likely translate to just a few words in another language.

The main equation describing IBM 1 and 2 is the evidence lower-bound(ELBO) shown in Eq. 5, and for the general case[1] its relationship to the log-likelihood is shown in Eq. 4. As a common issue with continues latent variables models, log-likelihood is infeasible both to compute and optimize as θ latent variable require integration over the whole space of possible values. Therefore, we use ELBO, which is both computable and possible to optimize, and introduce an approximate posterior with mean-field assumption $q = q(\theta, a^m|\phi,\lambda) = \prod_e^{|E|} q(\theta_e|\lambda_e) \prod_{j=1}^{m} q(a_j|\phi_j)$. In order to arrive at optimization equations 6 and 7 that mimic E and M steps respectively in EM algorithm [Dempster et al., 1977], we apply *functional derivatives* [Bishop, 2006] to Eq. 5, and ignore p(a—m,n) as it's a constant in the case of IBM1. In our notation $\mathbb{1}$ is identity function, and ψ is digamma function. The algorithm works as follows: we compute expected counts using Eq. 6 instead of true posterior as we did previously in E-step, and then compute parameters in Eq. 7 as M-step.

$$\log p(x) = ELBO - D_{KL}(q(z)||p(z|x)) \quad (4)$$

$$ELBO(f^m, e^n; \phi, \lambda) = \sum_{j=1}^{m} \left[E_q \left[\log \frac{p(a_j|m,n)}{q(a_j|\phi_j)} \right] + E_q \left[\log p(f_j|e_{a_j}, \theta_{f_j|e_{a_j}}) \right] \right] - \sum_{e}^{|E|} D_{KL} \left[q(\theta_e|\lambda_e)||p(\theta_e|\alpha_e) \right] \quad (5)$$

[1] The general case assumes that we have a model of data where x is observed random variable, and z is latent

$$q(a_k|\phi_k) = \frac{\exp(\psi(\lambda_{f_k|e_{a_k}}) - \psi(\sum_{f=1}^{|F|}\lambda_{f|e_{a_k}}))}{\sum_{i=0}^{n}\exp(\psi(\lambda_{f_k|e_{a_k}}) - \psi(\sum_{f=1}^{|F|}\lambda_{f|e_{a_k}}))} \quad (6)$$

$$\lambda_{f|k} = \alpha_f + \sum_{j=1}^{m} E_{q_\phi}\left[\mathbb{1}_k(e_{a_j})\mathbb{1}_f(f_j)\right] \quad (7)$$

2.3 IBM 2

IBM model 2 extends IBM model 1, by parameterizing the alignment distribution. Instead of assuming all alignments are equally likely, the model learns how likely each alignment is. A clever way of parameterizing this is using a jump function. The jump function measures the relative jump size between two positions in sentences of potentially different lengths. The jump function is shown in Eq. 8.

$$\delta(a_j = i, j, m, n) = i - \lfloor j\frac{m}{n} \rfloor \quad (8)$$

The alignment probabilities can then be parameterized as a categorical distribution over the δ values ranging from some [`-max_jump`, `max_jump`]. The alignment probability becomes $P(a_j = i \mid m, n) = Cat(\delta(a_j = i, j, m, n) \mid \lambda)$. A downside of this approach is that we lose all ordering between the jumps, i.e. a there is no link between a jump of size 2 and one of size 3. The incomplete-data likelihood and posterior distribution remain the same as in Eq. 1 and 2 for the IBM 1 model, except for that the alignment probabilities are no longer uniform. The EM algorithm now also requires an M-step for the alignment parameters. These will have the same form as those for the lexical translation parameters. The update equation for the alignment parameters is shown in Eq. 9.

$$\lambda_{\delta_k} = \frac{\mathbb{E}[\delta_k|A_1^n, \lambda_{\delta_k}]}{\sum_{l=-max\_jump}^{max\_jump}\mathbb{E}[\delta_l \mid A_1^n, \lambda_{\delta_t}]} \quad (9)$$

The expectations in equation 9 are with respect to the posterior distribution. Thus, essentially we are counting the expected number of times that an certain jump occurs in the parallel corpus weighted by the posterior probability in the E-step, while keeping parameters λ_{δ_k} fixed. In the M-step the λ_{δ_k} are updated by normalizing over the expected counts of all possible jumps.

A problem in the IBM 2 model is that there are no longer only global optima due to the mixture weights not being fixed anymore. We could, for example, swap around the component distributions of two mixture components. In IBM model 1 that would leave the likelihood unchanged, bringing it to another, but equally good, maximum. In IBM 2 this no longer holds, since the mixture weights are not necessarily uniform anymore. Therefore, one optimum can be better than another, and we are not guaranteed to reach a global optimum using the EM algorithm anymore. This means that it matters how the algorithm is initialized, in order to get to a good optimum. One trick that is often applied is to initialize the IBM 2 lexical translation parameters with ones obtained by running the IBM 1 model.

One other problem with the IBM 2 model discussed above, is that aligning words to the `NULL` token can cause unnaturally large jumps (to the beginning of the sentence) and might skew the jump distribution. These jumps are not representative of the data, but simply of the way the model has been engineered. We therefore add a special jump parameter to the categorical alignment distribution that represents jump to the `NULL` word. This parameter does not act in any special way, but simply acts as if it were another δ value.

2.4 IBM 2 Variational Bayes

Similarly to IBM1 with a prior over categorical parameters, we can extend IBM2 with a prior. Only E-step derivation that we've been using in IBM1 will have to be changed, in order to account for the alignment probability that we could previously ignore in the case of IBM1. Concretely, our previous Eq. 6 has to be replaced with Eq. 10 where $p(a_k|m, n)$ models the probability of alignment.

$$q(a_k|\phi_k) = p(a_k|m,n)\frac{\exp(\psi(\lambda_{f_k|e_{a_k}}) - \psi(\sum_{f=1}^{|F|}\lambda_{f|e_{a_k}}))}{\sum_{i=0}^{n}\exp(\psi(\lambda_{f_k|e_{a_k}}) - \psi(\sum_{f=1}^{|F|}\lambda_{f|e_{a_k}}))} \quad (10)$$

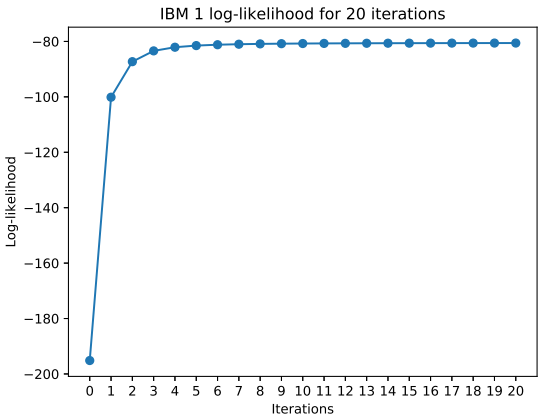
3 Experiments

In this section we evaluate the IBM 1 and 2 models, both their maximum likelihood versions and

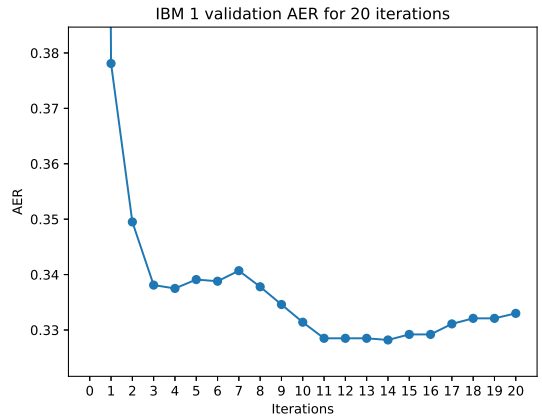
their variational versions. We use a parallel corpus taken from the Canadian Hansards parliament proceedings. We split up the data into three parts acting as training, validation and testing sets. In the experiments we'll compare the different models and experiment different settings within the models. We report the Alignment Error Rate (AER) on the test set for the best models of each type. An overview of all test results can be found in Table 1.

3.1 IBM 1

We ran the IBM 1 model using EM on the training data for 20 iterations. We track the log-likelihood normalized by the number of data points, and the Alignment Error Rate (AER) on the validation data. The AER is a number between 1.00 and 0.00, where lower is better. The results are shown in Figure 2. We omit the 0_{th} iteration for the AER plot, since it's 1.00.



(a) Log-likelihood



(b) Validation AER

Figure 2: The results of training for IBM 1 for 20 iterations.

We can see that as the likelihood converges and is always increasing as expected. We've picked two models to run on the test set, the first being simply the most trained model, i.e. the model for iteration 20, having the highest likelihood on the training data. The second model we pick is based on the validation AER, which was at its lowest point at 0.3282 in iteration 14. The results on the test set for these models were an AER of **0.3219** and **0.3217** for the training log-likelihood and validation AER criteria respectively.

3.2 IBM 1 Variational Bayes

To simplify our search space, we used a uniform hyper-parameter α, and ran the model with 4 different alpha configurations for 10 epochs in order to select the best one based on validation AER, as shown in Figure 3. After the hyper-parameter has been selected, which happened to be $\alpha = 1e-2$, we ran the model for 10 additional epochs, chose two based on ELBO and validation AER, and then evaluated them on the test set, as shown in Figure 4. As one can observe, the ELBO continues to improve on every step, which suggests that we should pick the model on 20th iteration based on the ELBO criteria. On the other hand, the validation AER flattens between epoch 11 and 14, and then starts to increase, it suggests that we can pick any model between 11 and 14 (inclusively), so we pick the one after 14th iteration. After we evaluated our selected models on the test dataset, we've got the following results: **0.327** and **0.328** AER for models selected based on validation AER and ELBO respectively. The difference is insignificant, and we can can't see that the model learns anything special after the 14th iteration.

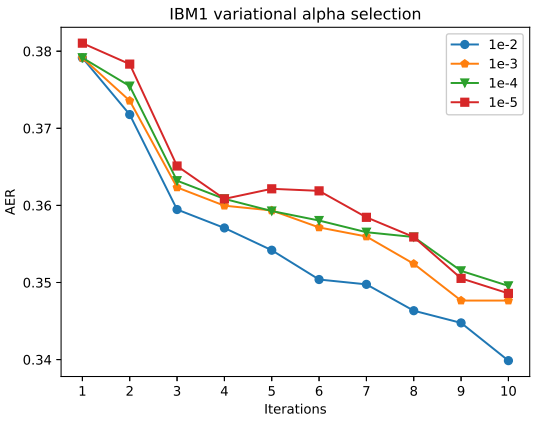
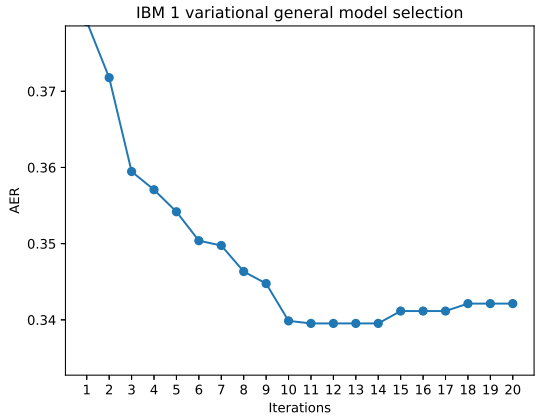
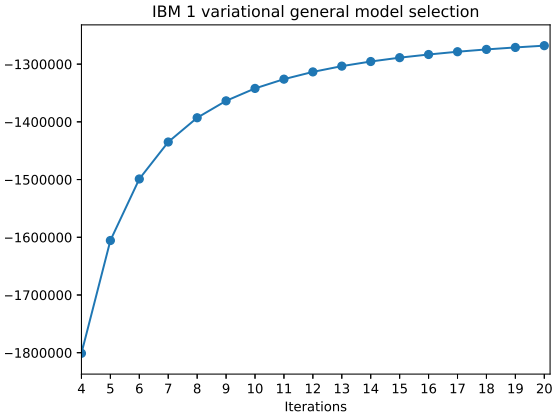


Figure 3: Validation AER for IBM1 variational models with 4 different alpha hyper-parameters.



(a) Validation AER.



(b) Training ELBO starting from 4th iteration to make the increase more visible.

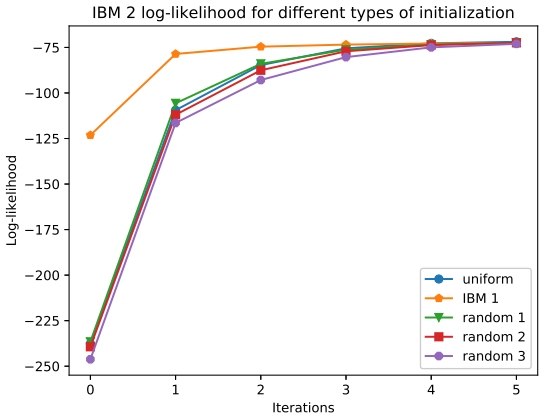
Figure 4: Performance of IBM 1 variational model trained for 20 epochs with $\alpha = 1e-2$ based on training ELBO and validation AER.

3.3 IBM 2

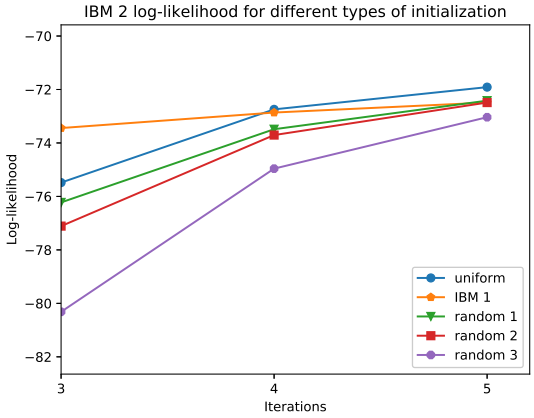
We experiment with three different types of initialization for the IBM 2 model: uniform initialization of all parameters, using the output of running IBM 1 for initializing the lexical parameters and initializing the jump parameters uniformly, and finally initializing all parameters randomly. For the random initialization we run experiments for three different runs. The results are shown in Figure 5 and 6.

The best model that comes out of this in terms of both log-likelihood and validation AER is the uniformly initialized one. This model we've ran for 20 iterations to see the further development of the log-likelihood and validation AER. The results of this are shown in Figure 7.

We've again selected two models two run on the test set, the most trained one with the lowest training log-likelihood, and the one with the best validation AER. The best validation AER was achieved on iteration 8 with a value of 0.2443. The AER on the test set for these two models were **0.2410** and **0.2395** respectively.



(a) Log-likelihood



(b) Zoomed in on the last two iterations

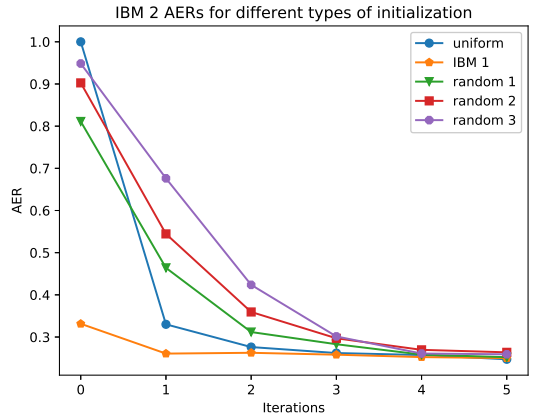
Figure 5: Log-likelihood of three different types of initialization on IBM model 2

3.4 IBM 2 Variational Bayes

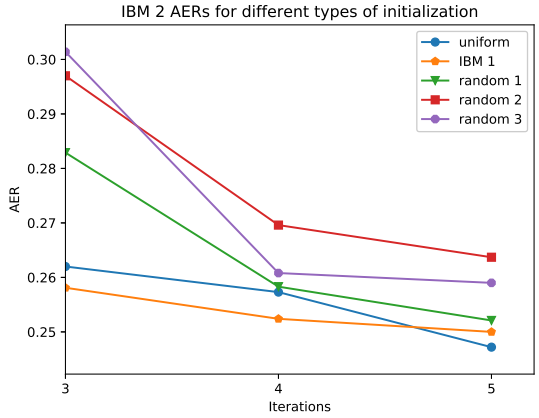
We also experimented with variational IBM 2 by pre-loading parameters from IBM 1 with $\alpha = 1e-2$ (using the same value for training). The results in terms of AER and ELBO are shown in Figure 8a and 8b respectively. The model's test AER is **0.231**, which is lower than the classical IBM's 2, which indicates that the prior is indeed beneficial.

4 Discussion

As we summarize in Table 1, the general results for IBM 1 and IBM 2 follow our intuition, IBM 2 in general performs better than IBM 1, be-

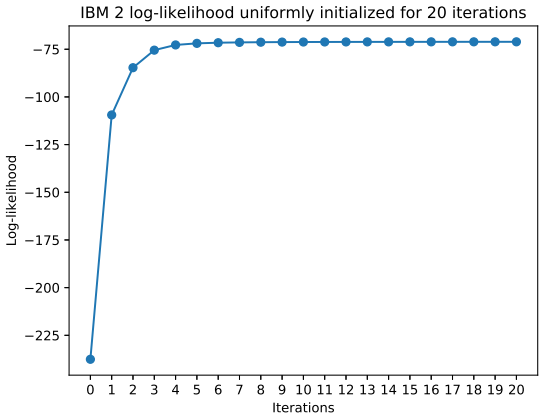


(a)

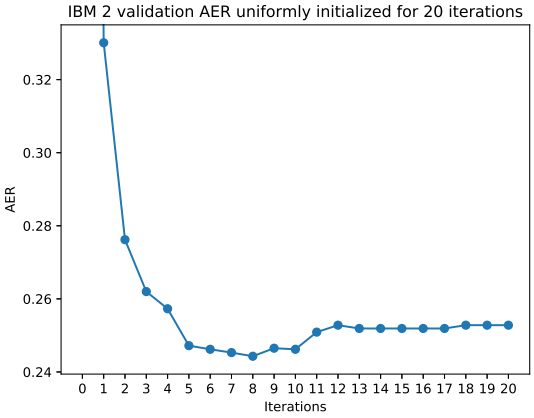


(b) Zoomed in on the last two iterations

Figure 6: AERs of three different types of initialization on IBM model 2



(a) Log-likelihood



(b) AER

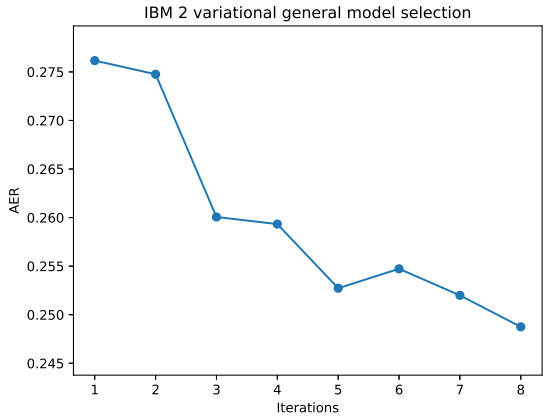
Figure 7: The results training the IBM 2 model for 20 iterations using uniform initialization

cause of the increased flexibility of the model. For IBM 2 the initialization mattered, mainly for the speed of convergence, but also for the final results. The uniform initialization somewhat surprisingly performed better than initializing the model with IBM 1 parameters by a small margin, although the model initialized with IBM 1 parameters converged faster. We think it might be coincidence that the uniform setting lead to a better optimum. Also, it might well be that another initialization setting would have performed better when run for 20 iterations than the uniform setting. In addition, we observed that both the IBM 1 and 2 variational models are sensitive to the choice of hyperparameter vector α, which suggests that there could be a better choice of α. For example, from the beginning we've made a simplifying assumption that $alpha$ vector is uniform, which could be slightly non-realistic because some words are likely to have multiple meanings and thus translate to several words, while other have only one meaning, and thus translate to exactly one word.

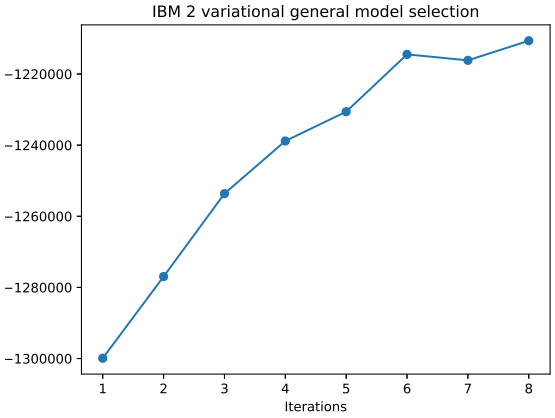
Model	**Test AER**
IBM 1	0.3217
Variational IBM 1	0.327
IBM 2	0.2395
Variational IBM 2	0.231

Table 1: Results of different models on the test set.

One trend in our experiments is that the validation AER seems to be the best predictor for the performance on the test set. Intuitively, performance on left-out data should be a better indicator of general performance than any measure on the testing data. Also, the log-likelihood on the testing data always increases or remains at the same value, so it can train as long as it can on the training data and is bound to overfit at some point.



(a) Validation AER.



(b) Training ELBO.

Figure 8: IBM2 variational performance over 8 epochs with initialization from IBM 1 variational with $\alpha = 1e - 2$.

During the experiments we noticed that ELBO's value in Eq. 5 is dominated by the Kullback-Leibler (KL) divergence part, which is independent of the dataset. Therefore, minimization of this KL should give the biggest increase of ELBO. However, it is yet unclear how much the distance between distributions actually affects the empirical results, and if a model with very small KL has any guarantees with respect to AER. Those questions are subjects of future work.

We also noticed that variational IBM model 1 does not have any edge in comparison to the classical one, it even performs slightly worse than the classical one. On the other hand, variational IBM model 2 outperforms the classical one. We hypothesize that it could be related to the fact that IBM 2 is a more complex model, and it's more likely to overfit, and it benefits more from the Dirichlet prior that forces integration over the whole space of parameters.

5 Conclusion

In our research we performed experiments with IBM 1 and 2 models both with and without a prior over categorical parameters. We conclude that IBM 2 outperforms IBM 1 based both on log-likelihood and AER, and that the prior is beneficial for IBM 2 as it potentially prevents it from overfitting. Furthermore, we have seen that the IBM 2 model does not have a single global optimum and that an algorithm such as EM is sensitive to the initial parameter settings. We also conclude that variational models are sensitive to the hyperparameter vector α, and a good choice of it can yield noticeably better results.

References

C. M. Bishop. Pattern recognition. *Machine Learning*, 128:1–58, 2006.

P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311, 1993.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.