# 1 Implementation

In this section we present our implementation of LDA in *Python* and discuss our results on real data.

## 1.1 Library Used

- *scikit learn* for pre-processing.
- *scipy* for probability distributions.

## 1.2 Dataset Description

The dataset used is the same one used in the authors implementation. it can be found at the following link: `https://nlp.stanford.edu/software/tmt/tmt-0.4/` It consist of 2246 small documents. We used 2000 documents for training our model and kept the rest for testing (classification).

## 1.3 Pre-processing

**Tokenization of the documents:** LDA uses the bag of word assumption (the order of words apparitions in a document is ignored). This allows us to use a compact representation of documents in a matrix of M x V. Each line in the matrix represents a document and the values are integer that represent the number of occurrences of a word in the vocabulary. This is similar to the one hot encoding trick that we used in the class.

In addition to tokenization, we included the following filters:

- Removed non-alphabetical characters.
- Removed words with only 1 occurrence across the corpus.
- Removed words that occur in 95 % or more of the documents.

Finally, Due to the limited memory of our hardware and for runtime reasons, we limited the vocabulary to only the 3000 most frequent words of the corpus.

## 1.4 Hyperparameter Selection

We chose our topic vector $\alpha$ to be of size 100 to match the original implementation of the author.

### 1.5 Parameter Initialization

As stated in section xxxx, LDA with with variational EM has 4 sets of parameters to estimate. The parameters initialized to the following value:

| Parameter | Dimension | Initialization |
|---|---|---|
| $\alpha$ | [K] | np.random.gamma(shape=np.ones((K)), scale=1/K) |
| $\beta$ | [K x V] | np.random.dirichlet(np.ones(V), K) |
| $\gamma$ | [M x K] | alpha + (N[d] / K) |
| $\phi$ | [M x N[d] x K] | np.ones((N[d], K))/ K |

### 1.6 Stopping Criteria

In the paper, the author doesn't state very clearly the stopping criterion used in its implementation. So we chose to stop the model once the parameter L2 norm of $\gamma$ from one iteration to the next is is below a given threshold.

## 2 Empirical Results

### 2.1 Topic Extraction

The canonical use of topic modeling is to find a list of topics across a corpus of text. We can attempt to understand the meaning of a topic using the words with the highest probability for the given topic.

Table 1: Topic Sample from LDA model: each column represents the top 10 words from the topic. We can see that the topics are easy to interpret. for full list of topic, see:

| Topic Sample | | | | |
|---|---|---|---|---|
| Topic 1: Politics | Topic 2: Finance | Topic 3: Trial | Topic 4: Crime | Topic 5: Korea |
| trade | york | years | authorities | roh |
| committee | shares | guilty | city | hostages |
| senate | board | case | injured | guard |
| new | million | trial | shot | officials |
| time | exchange | prison | hospital | coast |
| abortion | trading | said | night | korean |
| bush | stocks | judge | killed | korea |
| president | index | court | people | south |
| souter | market | attorney | police | north |
| said | stock | charges | said | said |

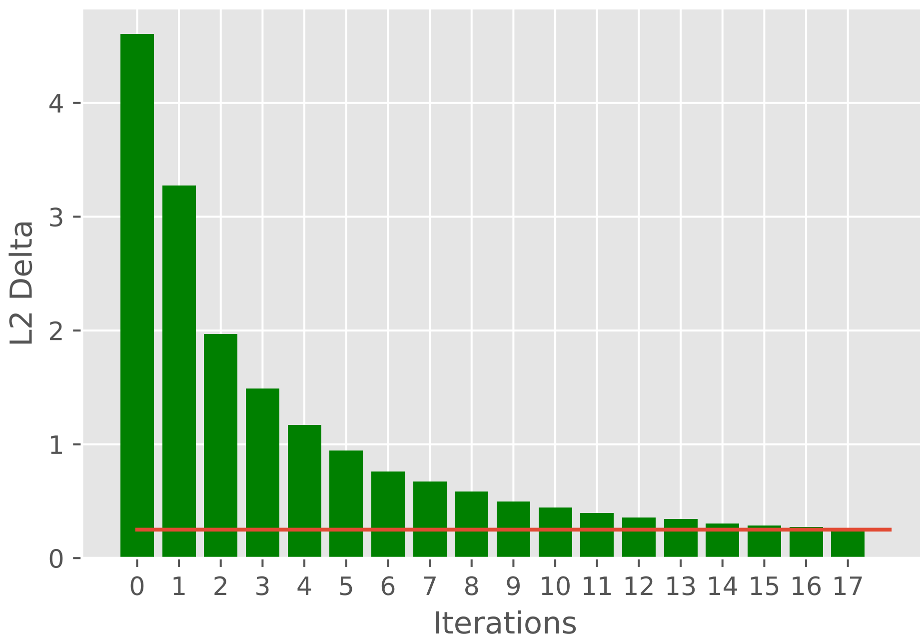


Figure 1: Change in $\gamma$ parameter across EM iterations. The green bars represent the L2 change of $\gamma$ parameters. The red line represent the stopping criterion used (0.25 in our case).

### 2.2 Document Classification

LDA can also be used to classify previously unseen document to one of the K-topics. First, we need to perform variational inference on the unseen document and use the trained gamma parameter to assign the document to the most likely topic. We scored the holdout documents and made a manual comparison of document vs topics. Overal, the topic made sense. Full results of the test set can be found in the following file:

`classification_holdout.csv`