# Report of Deep Learning for Natural Language Processing

Shixiang Li BY2203043

lishixiang@buaa.edu.cn

## Abstract

This report explores the use of the Latent Dirichlet Allocation (LDA) model for text classification. It first provides an overview of the fundamental principles of the LDA model and then details the process of applying LDA for text classification. Three experiments were conducted: comparing classification performance under different numbers of topics, analyzing the impact of using words versus characters as basic units, and evaluating classification performance across varying paragraph lengths. The experimental results indicate that increasing the number of topics appropriately, using words as basic units, and extending paragraph length can significantly enhance classification performance. Overall, the LDA model proves to be an effective approach for text classification, and fine-tuning relevant parameters can further improve classification outcomes.

## Introduction

### LDA Model

Latent Dirichlet Allocation (LDA) is a probabilistic graphical model used for topic modeling [1]. It is designed to uncover the hidden thematic structure within a collection of documents. LDA assumes that each document can be represented as a mixture of different topics, while each topic is characterized by a distinct distribution of words. By analyzing the words appearing in documents and inferring their associated topics, LDA effectively reveals the underlying thematic information within textual data.

The fundamental idea of LDA is that for a given document collection, each document is assumed to be generated by randomly selecting topics from a set of topic distributions and then randomly choosing words from the word distribution of each topic. In other words, LDA models the document generation process as a two-level stochastic process, where the content of a document is determined by topics, and the topics are in turn defined by words.

In practical applications, LDA is widely used for tasks such as text classification, topic analysis, and information retrieval. The typical workflow for using the LDA model includes data preprocessing (e.g., removing stop words, stemming), constructing a bag-of-words model or TF-IDF matrix, specifying the number of topics, training the model, and interpreting and evaluating the results.

LDA is widely applied in the field of natural language processing and serves as a powerful tool for understanding the thematic structure within large-scale textual data.

## Methodology

### Data Preprocessing

1. Import the corpus files.
2. Remove stop words from the corpus (optional).
3. Randomly sample 1,000 paragraphs from the corpus as the dataset, ensuring each paragraph contains K tokens.
4. Shuffle the sampled dataset randomly and split it into a training set and a test set in a 9:1 ratio.

### LDA Modeling and Text Classification Using SVM Classifier

1. Specify the number of topics as T, and train the LDA model using the preprocessed training set as the training samples.
2. Train a linear SVM classifier using the topic distributions and feature vectors from the training samples.
3. Use the trained classifier to obtain predicted labels for the training samples and compare them with the true labels to calculate the text classification accuracy.

### Analysis of the Impact of Parameters on Results

1. Compare the changes in classification performance under different settings of the number of topics, T.
2. Compare the classification performance differences between using "words" and "characters" as the basic units.

3. Compare the changes in classification performance with different settings of the token count, K, in the paragraphs.

## Experimental Studies

### Analysis of the Impact of the Number of Topics (T)

Experimental Setup: K=1000; Compare the changes in classification performance with different numbers of topics, T = 10, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500. The experimental results are shown in **Table I**:

**Table I Results of Different Number of Topics**

| No | Topic | Unit | Tokens | Stop Words | Acc.(%) |
|---|---|---|---|---|---|
| 1 | 10 | Word | 1000 | Y | 18.4 |
| 2 | 50 | Word | 1000 | Y | 47.8 |
| 3 | 100 | Word | 1000 | Y | 44.5 |
| 4 | 150 | Word | 1000 | Y | 33.6 |
| 5 | 200 | Word | 1000 | Y | 56.5 |
| 6 | 250 | Word | 1000 | Y | 44.5 |
| 7 | 300 | Word | 1000 | Y | 45.6 |
| 8 | 350 | Word | 1000 | Y | 43.4 |
| 9 | 400 | Word | 1000 | Y | 59.7 |
| 10 | 450 | Word | 1000 | Y | 52.2 |
| 11 | 500 | Word | 1000 | Y | 51.1 |

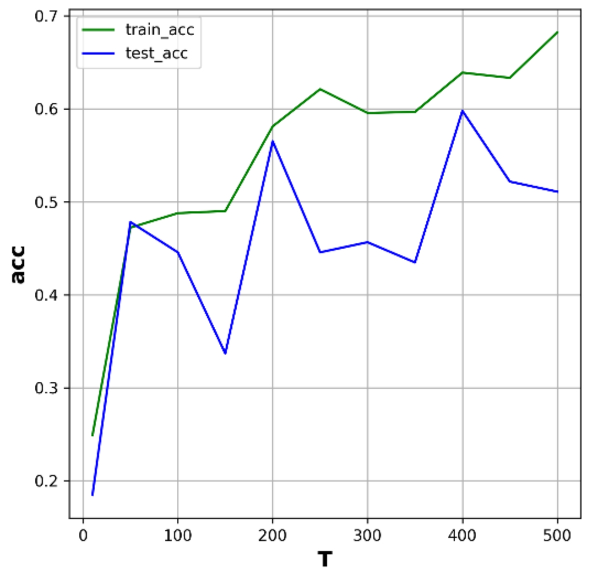

**Figure 1 Training loss curve of experiment I**

## Analysis of the Impact of Different Basic Units

Experimental Setup: Paragraph length K = 1000, number of topics T = 200; compare the classification performance differences between using "words" and "characters" as the basic units. The experimental results are shown in **Table II**:

**Table II Results of Different Basic Units**

| No | Topic | Unit | Tokens | Stop Words | Acc.(%) |
|---|---|---|---|---|---|
| 1 | 200 | Char | 1000 | Y | 56.5 |
| 2 | 200 | Word | 1000 | Y | 61.2 |

## Analysis of the Impact of Token Count

Experimental Setup: T = 200; compare the changes in classification performance with different token counts, K = 20, 100, 500, 1000, 3000. The experimental results are shown in **Table III**:

**Table III Results of Different Token Count**

| No | Topic | Unit | Tokens | Stop Words | Acc.(%) |
|---|---|---|---|---|---|
| 1 | 200 | Word | 20 | Y | 27.1 |
| 2 | 200 | Word | 100 | Y | 38.0 |
| 3 | 200 | Word | 500 | Y | 44.5 |
| 4 | 200 | Word | 1000 | Y | 59.7 |
| 5 | 200 | Word | 3000 | Y | 68.4 |

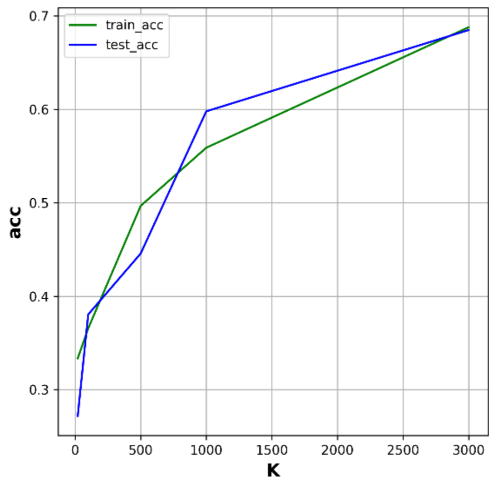

**Figure 2 Training loss curve of experiment Ⅱ**

# Conclusion

The following conclusions can be drawn from the experiments: 1) Increasing the number of topics appropriately can significantly improve classification performance; however, beyond a certain point, performance improvement is limited due to overfitting. 2) Classification performance using words as the basic unit is significantly better than using characters as the basic unit. 3) Increasing paragraph length helps significantly improve classification performance.

## References

[1] Wikipedia contributors. (n.d.). Latent Dirichlet allocation. Wikipedia. Retrieved March 23, 2025, from https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation