Reproducing FastText Language Identification

Including Kazakh Language Support

Ruslan Khissamiyev

Macquarie University

October 8, 2024

Outline

Introduction

In this presentation, we describe the reproduction of the language identification task using fastText, with an extension to include the Kazakh language. This work follows the tutorial by Edouard Grave, based on the original paper "Bag of Tricks for Efficient Text Classification" by Joulin, Grave, Bojanowski, and Mikolov (2017).

Downloading fastText

We downloaded the fastText library version 0.9.2 from GitHub:

```
# Downloading fastText
!wget https://github.com/facebookresearch/fastText
    /archive/v0.9.2.zip
!unzip v0.9.2.zip
```

Output (truncated for brevity):

```
creating: fastText-0.9.2/
creating: fastText-0.9.2/crawl/
inflating: fastText-0.9.2/crawl/README.md
...
```

Building fastText

We moved to the fastText directory and built it using `make`:

```
# Moving to the fastText directory and building it
%cd fastText-0.9.2
!make
```

Output (truncated for brevity):

```
c++ -pthread -std=c++11 -march=native -O3 -funroll-loops -
    DNDEBUG -c src/args.cc
...
```

Testing fastText

After building, we tested the fastText executable to ensure it was working:

```
# Testing if fastText is working
!./fasttext
```

Output:

```
usage: fasttext <command> <args>

The commands supported by fasttext are:

  supervised                  train a supervised classifier
  quantize                    quantize a model to reduce the memory
       usage
  test                        evaluate a supervised classifier
  test-label                  print labels with precision and
      recall scores
  predict                     predict most likely labels
  predict-prob                predict most likely labels with
      probabilities
  skipgram                    train a skipgram model
  cbow                        train a cbow model
  ...
```

Downloading the Data

We downloaded the sentences dataset from Tatoeba:

```
# Go back to the parent directory
%cd ..

# Downloading files from Tatoeba
!wget http://downloads.tatoeba.org/exports/
    sentences.tar.bz2
!bunzip2 sentences.tar.bz2
!tar xvf sentences.tar
```

Output (truncated for brevity):

```
--2024-10-09 10:18:21--  https://downloads.tatoeba.org/exports/
    sentences.tar.bz2
Resolving downloads.tatoeba.org (downloads.tatoeba.org)...
    94.130.77.194
Connecting to downloads.tatoeba.org (downloads.tatoeba.org)
    |94.130.77.194|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 195102325 (186M) [application/octet-stream]
Saving to: 'sentences.tar.bz2'
...
```

Preparing the Data

We prepared the data for fastText by reformatting and shuffling it:

```
# Preparing the data for fastText
!awk -F"\t" '{print "__label__"$2" "$3}' sentences
    .csv | shuf > all.txt
```

Splitting the Data

We split the data into training and validation sets:

```
# Splitting the data into training and testing
    sets
!head -n 10000 all.txt > valid.txt
!tail -n +10001 all.txt > train.txt
```

Training the Initial Model

We trained the initial model:

```
# Training the model
!fastText-0.9.2/fasttext supervised -input train.
    txt -output langdetect -dim 16
```

Output:

```
Read 100M words
Number of words:  4556997
Number of labels: 420
Progress: 100.0%  words/sec/thread: 253825  lr: 0.000000  avg.
    loss: 0.138132  ETA: 0h 0m 0s
```

Evaluating the Initial Model

We tested the model:

```
# Testing the model
!fastText-0.9.2/fasttext test langdetect.bin valid
    .txt
```

Results:

```
N       10000
P@1     0.953
R@1     0.953
```

The initial model achieved an accuracy of approximately 95.3%.

Morphological Features of Kazakh Language

- **Language Family**: Kazakh is a Turkic language, part of the Altaic family.
- **Agglutinative Morphology**:
  - Words are formed by adding a sequence of suffixes to roots.
  - Each suffix conveys specific grammatical meanings (e.g., tense, case, number).
- **Vowel Harmony**:
  - Vowels within a word harmonize to be front or back vowels.
  - Influences suffix selection and word formation.
- **Rich Inflectional System**:
  - Nouns inflect for multiple cases (e.g., nominative, genitive, dative).
  - Verbs conjugate for tense, mood, aspect, person, and number.
- **Comparison to English**:
  - English relies more on word order and auxiliary words.
  - Kazakh uses suffixes to express grammatical relationships.
- **Implications for NLP and fastText**:
  - Models must handle complex morphological structures.
  - Importance of subword features and character n-grams.

Utility of Adding Kazakh Language Support

- **Supporting Low-Resource Languages**:
  - Addresses the gap in NLP resources for Kazakh.
  - Promotes linguistic diversity in computational models.
- **Significant Speaker Base**:
  - Over 13 million native speakers worldwide.
  - Official language of Kazakhstan with growing digital presence.
- **Academic Importance**:
  - Enables research on agglutinative and morphology-rich languages.
  - Provides insights into handling complex linguistic features in NLP.
- **Practical Applications**:
  - Enhances language detection and processing in multilingual settings.
  - Supports development of educational tools and resources for Kazakh speakers.
- **Improving Multilingual Models**:
  - Increases robustness and generalization of NLP models.
  - Facilitates better cross-lingual understanding and transfer learning.
- **Contributing to Language Preservation**:
  - Aids in digital preservation efforts of the Kazakh language.
  - Encourages creation of digital content and resources in Kazakh.

Including Kazakh Language

We extended the model to include the Kazakh language (ISO code: `kaz`):

```
# Checking the number of Kazakh sentences
!awk -F"\t" '$2 == "kaz"' sentences.csv | wc -l
```

Output:

```
4335
```

We have 4,335 Kazakh sentences available.

Preparing Kazakh Sentences

We extracted Kazakh sentences and appended them to the dataset:

```
# Extracting Kazakh sentences
!awk -F"\t" '$2 == "kaz" {print "__label__"$2" "$3
    }' sentences.csv > kazakh_sentences.txt

# Appending Kazakh sentences to the dataset
!cat kazakh_sentences.txt >> all.txt

# Shuffling the dataset
!shuf all.txt -o all_shuffled.txt
```

Splitting the Updated Dataset

We split the updated dataset into training and validation sets:

```
# Use 10,000 samples for validation
!head -n 10000 all_shuffled.txt > valid.txt

# Use the rest for training
!tail -n +10001 all_shuffled.txt > train.txt
```

Retraining the Model with Kazakh

We retrained the model including Kazakh sentences:

```
# Retraining the model with Kazakh sentences
!fastText-0.9.2/fasttext supervised -input train.
    txt -output langdetect -dim 16
```

Output:

```
Read 100M words
Number of words:  4556997
Number of labels: 420
Progress: 100.0%  words/sec/thread: 191641  lr: 0.000000  avg.
    loss: 0.167677  ETA: 0h 0m 0s
```

Evaluating the Updated Model

We tested the updated model on the validation set:

```
# Testing the model on the validation set
!fastText-0.9.2/fasttext test langdetect.bin valid
    .txt
```

Results:

```
N       10000
P@1     0.954
R@1     0.954
```

The overall accuracy remained consistent at approximately 95.4%.

Evaluating on Kazakh Sentences

We specifically assessed the model on Kazakh sentences:

```
# Preparing Kazakh test set
!awk -F"\t" '$2 == "kaz" {print "__label__"$2" "$3
    }' sentences.csv | shuf > kazakh_test.txt

# Testing the model on Kazakh sentences
!fastText-0.9.2/fasttext test langdetect.bin
    kazakh_test.txt
```

Results:

```
N         4335
P@1       1
R@1       1
```

The model achieved 100% accuracy on Kazakh sentences.

Improving the Model with Hyperparameter Tuning

To enhance the model's performance, especially on the validation set, we experimented with hyperparameter tuning:

- **Increased Number of Epochs**:
  - Trained the model for 15 epochs to allow it to learn better representations.
- **Used Hierarchical Softmax**:
  - Changed the loss function to hierarchical softmax for efficiency.
- **Adjusted Learning Rate**:
  - Increased the learning rate to 1.0 for faster convergence.
- **Included Subword Features**:
  - Set `minn` and `maxn` to use character n-grams.
- **Used Word n-grams**:
  - Included word bigrams to capture local context.

Command:

```
# Retraining the model with all improvements
    applied
!fastText-0.9.2/fasttext supervised \
-input train.txt \
-output langdetect \
```

Evaluating the Improved Model

We tested the improved model on both the validation set and the Kazakh test set.

- **Validation Set Results**:

```
N       10000
P@1     0.976
R@1     0.976
```

- **Kazakh Test Set Results**:

```
N       4335
P@1     1
R@1     1
```

- **Observations**:
  - Overall validation accuracy improved from 95.4% to 97.6%.
  - Accuracy on Kazakh sentences remained at 100%.

Conclusion

By extending the fastText language identification model to include Kazakh and applying hyperparameter tuning, we achieved:

- Improved overall validation accuracy to 97.6%.
- Maintained high accuracy on Kazakh sentences at 100%.
- Demonstrated the effectiveness of hyperparameter tuning in enhancing model performance.