Language Identification: Comparing Machine Learning and Non-Machine Learning Approaches

Conor O`Sullivan
Trinity College Dublin
osullc43@tcd.ie

Ashutosh Sharma
Trinity College Dublin
sharmaas@tcd.ie

Yifang Wang
Trinity College Dublin
wangy12@tcd.ie

Xingyu Qiu
Trinity College Dublin
qiux@tcd.ie

***Abstract*:** Language identification involves determining the natural language of a given piece of content. In this paper, we compared the performance of an N-gram model, Artificial Neural Network (ANN) and Support Vector Machines (SVM), in identifying the language of text snippets from 6 languages. All three models achieved an accuracy of over 98% and the accuracy of the N-gram model was nearly 100%.

Keywords: Language Detection, N-gram, Artificial Neural Network, Support Vector Machine.

1 Introduction

We compare machine learning (ML) and non-machine learning (non-ML) approaches to predicting the language of short written text. The text comes from 6 languages: English, German, Spanish, French, Portuguese and Italian. For the non-ML approach we fitted an N-gram model. For the ML approach we fitted both an SVM and an ANN. We also compare these models against a simple baseline - predict the most common language in the test set.

The N-gram model had the best performance with an accuracy of close to 100%. This was 1.340 percentage points more than the SVM and 1.647 more than the ANN. We found all 3 models to perform significantly better than the baseline.

2 Related Work

Various approaches have been taken to solve the language identification problem. Ramisch (2008) used an N-gram model to predict the language of sentences from the European Union Parliament speeches. They took speeches in 5 languages: French, Portuguese, English, German and Finnish. The model had an accuracy of more than 99.9%. Note the similarity of the text size and languages in Ramisch (2008)'s paper and in this paper. This suggests we should expect similar accuracy when using an N-gram model.

Work has also been done in applying ML methods to the problem. Simões et al. (2014) applied an ANN to classify TED conferences in 25 different languages with 97% accuracy. In another paper, Gerrit et al. (2012) use an SVM to identify 11 different South African languages. For input strings of 100 characters they achieved an accuracy of 97%. The results of these papers cannot be compared directly to the N-gram model above as there are differences besides the models used (i.e. datasets and languages tested).

3 Methodology

3.1 Data

We used a dataset of 6,872,356 text snippets in 328 unique languages (Tatoeba.org, 2018). To simplify our problem we used:

- The 6 most popular written Latin languages on the internet: English, German, Spanish, French, Portuguese and Italian (W3techs.com, 2018).
- Text snippets between 20 and 200 characters long.
- 50,000 text snippets from each of these languages. That is 300,000 rows in total.

Table 1 shows examples of these text snippets.

Table 1: Examples of Text Snippets

Language	Text
English	He doffed his hat when he saw me.
German	Ich war um seine Gesundheit besorgt.
Spanish	El surgimiento del exoesqueleto en los artrópodos fue un acontecimiento evolutivo muy importante para esos animales.
French	Il a ce qu'il faut pour réussir dans le monde des affaires.
Portuguese	Cante-me uma canção de ninar.
Italian	Non lo augurerei a nessuno.

The data was divided into a training (70%), validation (20%) and test (10%) set. We performed tests to ensure no data leakage occurred. That is, we counted the number of sentences that appear in both the:

1. Training and validation sets
2. Training and test sets
3. Validation and test sets

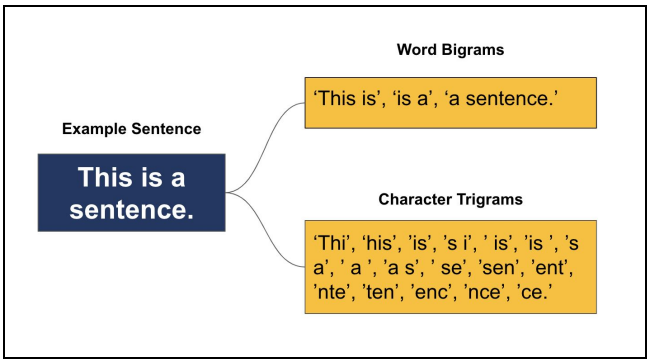
3.2 Baseline Model

The baseline model was a simple heuristic - always predict the most commonly occurring language in the test set. In this case it was English.

3.3 N-gram Feature Extraction

An n-gram is a sequence of n consecutive words/characters in a text. Figure 1 shows how a sentence can be broken down into word 2-grams (bigrams) and character 3-grams (trigrams).

Figure 1: Word Bigrams and Character Trigrams



3.4 N-gram Model

This model aims to determine the similarity of input text and the given languages using a measure called *perplexity*. The basic assumption is that each word depends only on the last $n-1$ words, i.e, $P(w_1 w_2 ... w_n) \approx \prod_i P(w_i \mid w_{i-k} ... w_{i-1})$.

We used a tri-gram model were $n = 3$.

We first separated the training dataset by language, tokenized them as lists of tri-grams with occurrence probability. Then, *perplexity* is calculated for the testing set:

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i \mid w_{i-2} w_{i-1})}}$$

The text snippet is classified as the language that gives the lowest *perplexity*. We implemented the model using *nltk* and *kenlm* package in Python.

3.5 Artificial Neural Network

The following was done to create features:

- Using the training dataset, we selected the 50, 100 and 200 most frequent character trigrams from each language.
- The unique trigrams from each list were taken, giving us 3 trigram sets.
- Using these, matrices of vectorised sentences were created for the training, validation and testing sets.

Table 2 shows an example of a feature matrix.

Table 2: Example of Trigram Feature Matrix

	_Ma	_To	_a_	...	ve_	you	ão_
0	1	0	0	...	0	1	0
1	0	2	0	...	0	0	0
2	0	0	1	...	1	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Multilayer perceptrons with 1 hidden layer were used. For each feature set, we tested 3 different models. Each model had a hidden layer with $3/4(N+R)$, $1/2(N+R)$ and $1/4(N+R)$ units respectively. Where:

- $N := \#number\ of\ features\ in\ the\ input\ layer$
- $R := \#number\ of\ languages\ in\ the\ output\ layer$

The summary of all the models is seen in Table 3.

Table 3: ANN Models Tested

Model	# Trigram	# Features	Units in Hidden layer
1	50	196	152
2			101
3			51
4	100	349	266
5			178
6			89
7	200	672	509
8			339
9			170

Model training was done using 50,000 training set rows and validated using 5,000 validation rows. The best model was then selected and trained using the entire training set. We used the TensorFlow python library (TensorFlow, 2018) to construct the models.

3.6 Support Vector Machines

We used SVMs with a linear kernel. The same feature sets as ANN model were used. We also varied the C hyper parameter. We used the following 6 values: 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}.

The models were trained using 50,000 training rows. The best model was then selected and trained using the entire training set. We used the sklearn SVM library to implement this model.

3.6 Metrics

We evaluate the final models using the test set and the following metrics:

- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix Heatmap

We used McNemar's test to determine the statistical significance of differences in the model's accuracies. To calculate the test statistic we first construct the matrix in Table 4.

Table 4: McNemar's Test Matrix

		Model 1	
		Correct	Incorrect
Model 2	Correct	N_{00}	N_{10}
	Incorrect	N_{01}	N_{11}

- N_{00} := *# times both models are correct*
- N_{10} := *# times model 1 is incorrect and model 2 is correct*
- N_{01} := *#times model 1 is correct and model 2 is incorrect*
- N_{11} := *# times both models are incorrect*

Then the test statistic is then given by:

$$\chi^2 = \frac{(N_{10} - N_{01})^2}{(N_{10} + N_{01})}$$

This statistic has a chi-squared distribution with 1 d.o.f. (Gillick, L. and Cox, S.J., 1989).

4 Results

4.1 Data Checks

We found no sentence appeared in more than one of the 3 datasets. I.e. there was no data leakage.

4.2 N-gram model

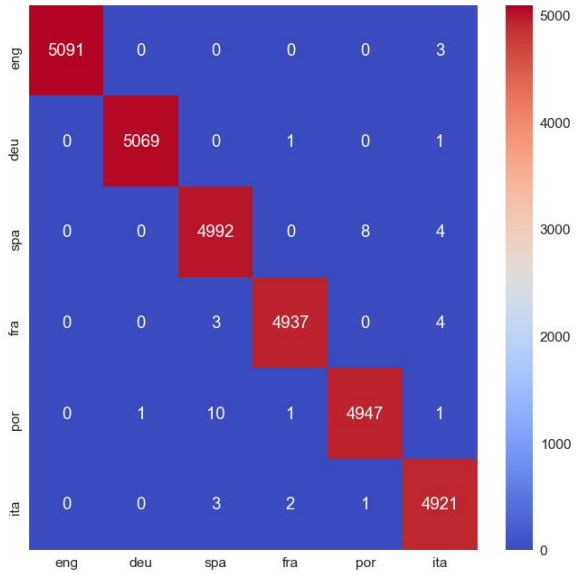
The model achieved an accuracy of 99.86%. Table 5 shows for each language the precision was above 99.7%.

Table 5. N-gram Model Precision, Recall and F1-Score

	Statistical N-gram Model			
Language	Precision	Recall	F1-Score	# of Records
English	100.0%	99.9%	100.0%	5,094
German	100.0%	100.0%	100.0%	5,071
Spanish	99.7%	99.8%	99.7%	5,004
French	99.9%	99.9%	99.9%	4,944
Portuguese	99.8%	99.7%	99.8%	4,960
Italian	99.7%	99.9%	99.8%	4,927
Average	99.9%	99.9%	99.9%	30,000

Figure 2 shows the misclassification cases among languages. The red diagonal indicates most of the cases were correctly classified.

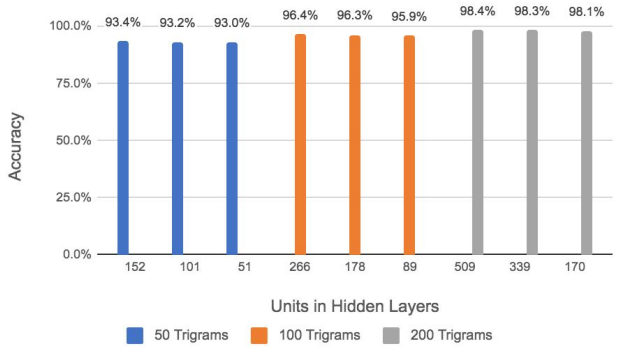
Figure 2. N-gram Model Confusion Matrix Heatmap



4.3 ANN

Figure 3 shows the 200 trigram feature set and 509 hidden layers produced the highest accuracy of 98.4%. This model was then refitted with all the training data.

Figure 3: Accuracy of Various ANN Models



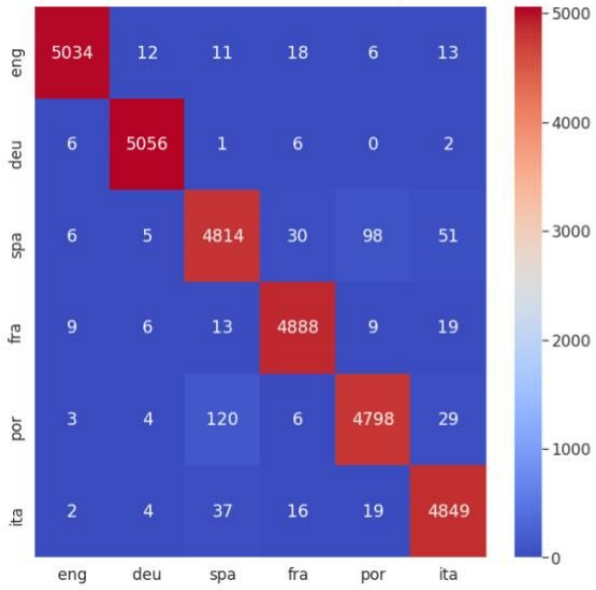
The model achieved an overall accuracy of 98.11%. Table 6 shows the model was the worst at predicting Spanish as this language had the lowest precision and recall.

Table 6: ANN Precision, Recall and F1-Score

	Artificial Neural Network			
Language	Precision	Recall	F1-Score	# of Records
English	99.5%	98.8%	99.2%	5,094
German	99.4%	99.7%	99.6%	5,071
Spanish	96.4%	96.2%	96.3%	5,004
French	98.5%	98.9%	98.7%	4,944
Portuguese	97.3%	96.7%	97.0%	4,960
Italian	97.7%	98.4%	98.1%	4,927
Average	98.1%	98.1%	98.1%	30,000

The red diagonal in Figure 4 indicates that the model classifies most of the testing set correctly. The relatively high values for the Spanish-Portuguese and Portuguese-Spanish cells suggest these languages are often mistaken for each other.

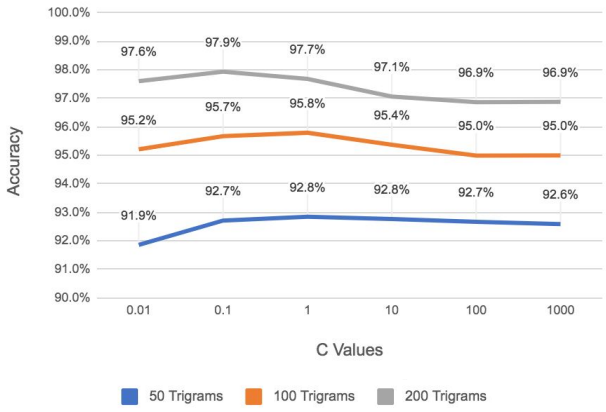
Figure 4: ANN Confusion Matrix Heatmap



4.4 SVM

From Figure 4, we can see that the 200 trigram feature set and C=0.1 had the highest accuracy of 97.9%. This SVM model was then refitted using all the training data.

Figure 4: Accuracy of Various SVM Models



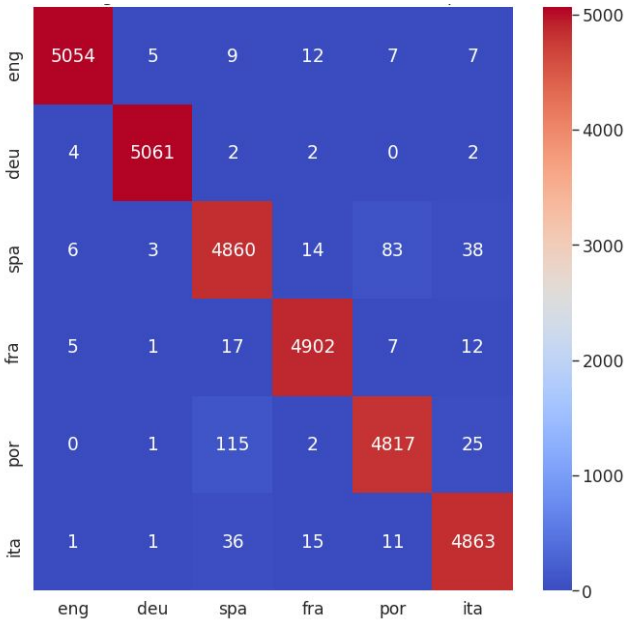
The final model achieved an accuracy of 98.52%. Table 7 shows the precision, recall and F1-Score of this model. It appears that this model classified Spanish the worst.

Table 7: SVM Precision, Recall and F1-Score

Language	Support Vector Machine			
	Precision	Recall	F1-Score	# of Records
English	99.7%	99.2%	99.4%	5,094
German	99.8%	99.8%	99.8%	5,071
Spanish	96.6%	97.0%	96.8%	5,004
French	99.0%	99.2%	99.1%	4,944
Portuguese	97.7%	97.2%	97.4%	4,960
Italian	98.3%	98.7%	98.5%	4,927
Average	98.5%	98.5%	98.5%	30,000

Figure 5 shows, like the ANN, Portuguese and Spanish are often mistaken for each other. This is likely due to the similarities of the two languages.

Figure 5: SVM Confusion Matrix Heatmap



4.4 Comparison

From Table 8 we see the accuracy of the N-gram model is 1.340 percentage points higher than the SVM and 1.647 points higher than the ANN. The ANN's accuracy is 0.407 points lower than the SVM. All the models outperformed the baseline.

Table 8: Summary of Accuracy

Model	Accuracy
Baseline	16.980%
ANN	98.110%
SVM	98.517%
N-Gram	99.857%

Table 9 shows that the results of the McNemar's tests. We see that all the accuracy differences are statistically significant. In other words, the N-gram model is statistically the best classifier followed by the SVM and then the ANN.

Table 9: Statistical Significance of Accuracy

Model 1	Model 2	Test Statistic	P-Value
Baseline	ANN	24295.22	<0.0001
Baseline	SVM	24379.27	<0.0001
Baseline	N-Gram	24855.00	<0.0001
ANN	SVM	23.00	<0.0001
ANN	N-Gram	445.14	<0.0001
SVM	N-Gram	349.57	<0.0001

In terms of computational cost, the N-gram model is also the easiest to build. Hence, we conclude that, within this context, N-gram model is the best classifier.

The high accuracy of the models was a concern. However, comparing our results to those of other researchers, such as Ramisch (2008), we can see these accuracy levels are to be expected.

5 Limitation and Outlook

We have identified the following limitations of our research:

- Our results do not necessarily extend to larger text snippets.
- We may not get similar results if we modelled more than 6 languages.
- It is not clear if these results would extend to non-latin languages.

We would like to expand results by either modelling more languages or modelling longer text snippets. We could construct an ensemble model of various models aimed at predicting different length text snippets.

REFERENCES

Ahmed, B., Cha, S.H. and Tappert, C., 2004. Language identification from text using n-gram based cumulative frequency addition. Proceedings of Student/Faculty Research Day, CSIS, Pace University, pp.12-1.

Botha, G.R. and Barnard, E., 2012. Factors that affect the accuracy of text-based language identification. *Computer Speech & Language*, *26*(5), pp.307-320.

Gillick, L. and Cox, S.J., 1989, May. Some statistical issues in the comparison of speech recognition algorithms. In

Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on (pp. 532-535). IEEE.

Ramisch, C., 2008. N-gram models for language detection. M2R Informatique-Double diplˆome ENSIMAG–UJF/UFRIMA.

Simões, A., Almeida, J.J. and Byers, S.D., 2014. Language identification: a neural network approach. In OASIcs-OpenAccess Series in Informatics (Vol. 38). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Tatoeba.org, (2018). Sentences.csv. Available at: https://downloads.tatoeba.org/exports/ [Accessed 16 October 2018]

TensorFlow. (2018). *Premade Estimators | TensorFlow*. [online] Available at: https://www.tensorflow.org/guide/premade_estimators [Accessed 25 Oct. 2018].

Mark Galea. 2016. Language Detection using N-Grams. [ONLINE] Available at: http://cloudmark.github.io/Language-Detection/. [Accessed 16 October 2018].

W3techs.com, (2018). Usage of content languages for websites. [online] Available at: https://w3techs.com/technologies/overview/content_language/all [Accessed 16 October 2018]

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NIPS 2015), pages 649–657, Montreal, Canada, December.