Topological Language Analysis and Language Classification using N-Grams

IMCS 3010U Final Report

https://github.com/RyanCoones/IMCS3010U-Project

Ryan Coones

December 2, 2025

Contents

1 Introduction

Languages are fascinating, complex, and organized systems of sounds used to express ideas, emotions, and messages. They exhibit patterns that create distinct sound profiles. Languages have evolved and influenced each other throughout history as a consequence of conquests, migrations, cultural exchanges, as well as through proximity and trade. This influence can impact the sound profiles of languages, cause an exchange of vocabulary, and even grammar structures. Because of this, linguists have categorized languages into groups based on their historical connections, grammar, lexicon and other linguistic structures. To name a few, we have Germanic languages (German, Dutch, English, Swedish), Romance languages (French, Italian, Portuguese, Romanian, Spanish), Sino-Tibetan languages (Mandarin, Cantonese, Tibetan, Burmese), etc.

In this report, we will examine the process and experimental findings regarding the use of n-gram frequencies to represent the orthographic structure of languages in order to topologically visualize these groups of languages, as well as classifying samples of texts in various languages. We will introduce some background knowledge, the methodology used, challenges faced throughout the process, methods of assessing/evaluating the results, the results themselves, as well as some suggestions for future development. This project is a compelling integration of mathematics and computer science, since it includes building multi-dimensional, numerical representations of linguistic data, applying mathematical techniques like cosine similarity and dimensionality reduction to reveal hidden patterns, as well as the utilization of machine learning methods to classify languages based on the extracted features. By embedding numerical representations of languages into a two-dimensional plot, I hope to illustrate how closely related they are based on their structural and statistical characteristics, both visually through clustering and quantitatively through classification accuracy.

2 Background

2.1 N-gram Frequency Vectors

- An n-gram is a contiguous sequence of n items, such as letters, syllables, or words, extracted from a given text or speech corpus, most commonly used for natural language processing tasks (NLP). [1]
- N-grams approximately represent the orthographic structure of languages.
- Mathematically, each language can be represented as a point in $\mathbb{R}^{|V_L|}$, where $|V_L|$ is the size of the n-gram vocabulary corresponding to language L.

2.2 Cosine Similarity and Distance

- Cosine similarity is used to measure how similar two languages are:

$$\text{sim}(A, B) = \frac{A \cdot B}{\|A\|\|B\|}$$

  Where A and B are vectors representing two languages' orthographic structure, and $\text{sim}(A, B)$ is the cosine of the angle between A and B.

- Distance is defined as $1 - \text{sim}(A, B)$, forming the basis of the distance matrix.

2.3 Dimensionality reduction (UMAP)

- UMAP (Uniform Manifold Approximation and Projection) projects high dimensional vectors into two-dimensional space. [2]
- The resulting axes are abstract dimensions that preserve patterns of similarity and local relationships.

2.4 Classification Models

- Using a multilayer perceptron (MLP) to perform multi-class classification on languages.
- Model takes a tensor of n-gram frequencies, and identifies which language it most closely resembles.
- Implemented using PyTorch Lightning

3 Challenges

3.1 Classification Complexity

- Initially unsure about complexity of language classification, and was considering binary classification or multi-class classification

3.2 Inconsistent size in n-gram vocabularies

- Raises concerns regarding validity/method of comparison.
- Causes issues picking a size for the input dimension of the neural network.

3.3 Transformer Efficiency

- Experimented with a transformer model to transcribe orthographic text corpora into IPA (International Phonetic Alphabet) text to get a more accurate representation of a language's phonetic structure. [3]
- Transcription of text was demanding on memory and time.
  - Limited number of lines to be transcribed to 1000 - Received an error about using too much memory.
  - Limited number of lines to be transcribed to 100 - This process ran for 20 minutes to transcribe a single language, and the output was unusable.
  - Transcribing after generating n-grams is also impractical, since some transcriptions vary in length, making n-gram comparisons inconsistent.
- Given these constraints, using a transformer for transcription proved impractical. The focus of the project shifted towards improving the clustering of languages based on orthographic text, and language classification.

4 Methodology

4.1 Data Preparation

- Corpus selection
- Text Preprocessing
  - Removed newline characters as they are not relevant to orthographic structure and are not contained in natural text, removed punctuation, and converted all text to lowercase.
  - Experimented with removal of diacritics - decided against this, as diacritics encode orthographic data about a language.
  - Experimented with different sizes of n-grams to observe the effectiveness in extracting meaningful linguistic data without losing generality of the orthographic structure.
  - Creation of global vocabulary

4.2 Language Profiles

- Constructed n-gram frequency vectors for each language - counted occurrences of each unique n-gram found in each languages corpora, computed ratio between occurrences and total vocabulary size.
- Represented each language as a high dimensional vector in $\mathbb{R}^{|V_L|}$, Where $|V_L|$ is the size of the n-gram vocabulary for the language L.

- Converted language vectors into vectors of dimension $|V_p|$ where $|V_p|$ is a pooled vocabulary of n-grams across all languages corpora to ensure consistent vector dimensions, enable valid comparisons as well as fix issue with input dimension of neural network.
  - When an n-gram is not contained in a language's corpora, the corresponding entry is 0. Otherwise, it is calculated as the ratio of the occurrences of the n-gram to the total vocabulary size.

4.3 Similarities and Differences

- Calculated similarity using cosine similarity metric.
- Converted similarity to distance $(1 - \text{similarity})$.
- Generated a symmetric distance matrix representing pairwise language differences.

4.4 Dimensionality Reduction

- Applied UMAP to project language vectors into two dimensional space.
- Used cosine similarity metric and n_neighbors = 2.
- Produced a visualization of language similarity on abstract axes.
- At first, the similarity plot was quite poor in showing clusters of language groups:

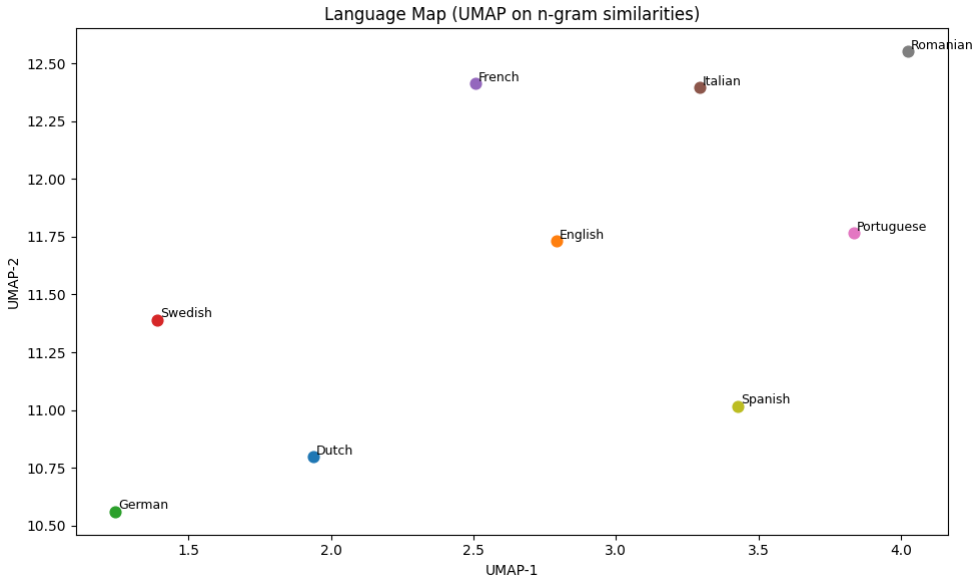

Figure 1: Original UMAP projection of language vectors (n_neighbors = 15).

Setting the UMAP reducer parameter of n_neighbors to 2 greatly exaggerated the clustering of the language groups. [4] By default, n_neighbors is set to 15, which is greater than the number of languages included in the dataset, which negatively affected clustering performance. Generally, a smaller value of n_neighbors highlights local relationships, producing tighter clusters at the cost of global accuracy, which is ideal given the small number of languages in this dataset. Below are the resulting plots after making this adjustment.

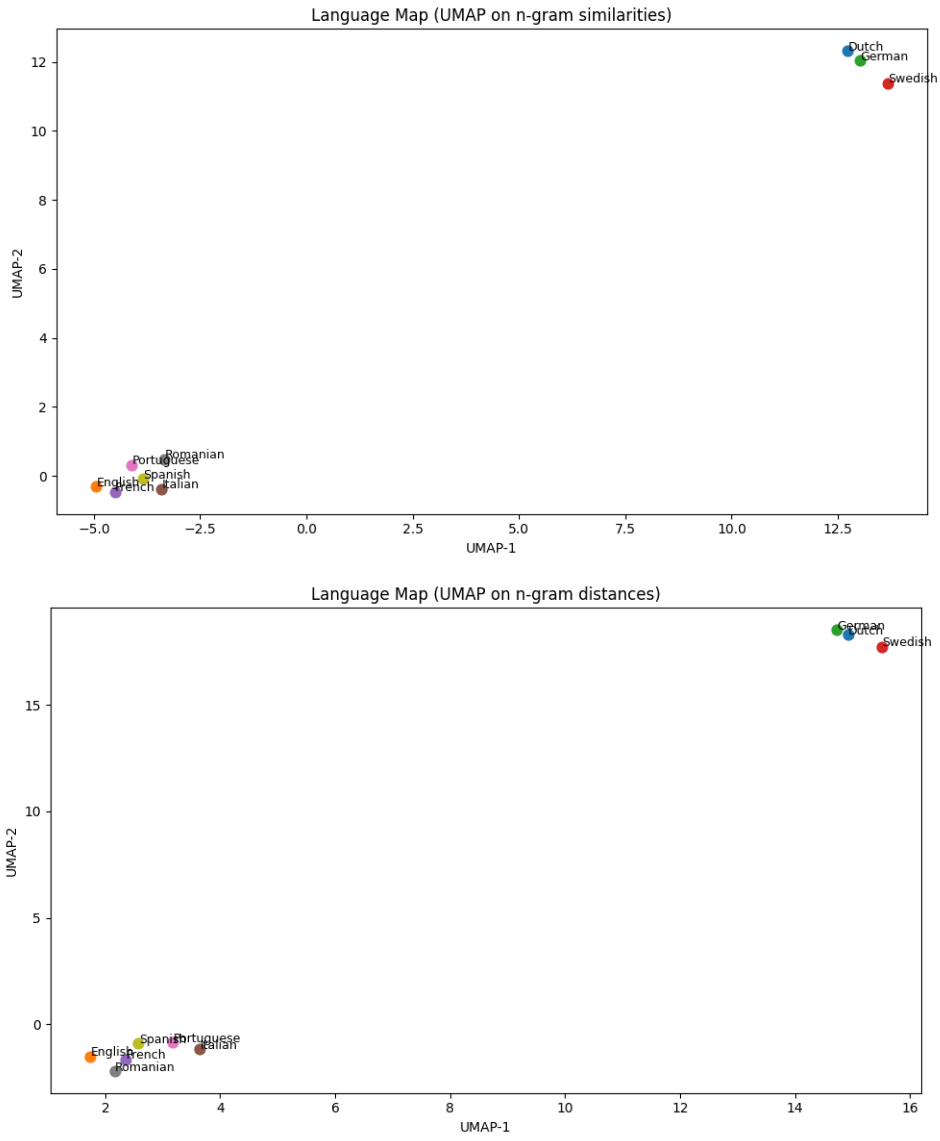

Figure 2: Improved UMAP projection of language vectors (n_neighbors = 2).

4.5 Language Classification

- Using a multi-layer perceptron built with PyTorch Lightning to perform multi-class classification to identify languages from a sample text.
- Definition of Lightning Module requires defined methods for forward propagation, the training step, and to configure an optimizer.
- Lightning Modules typically involve a validation step, but we ignore this step, as we are not tuning any hyper-parameters, and we are cross validating after training the model. More on this later.
- Model Architecture:
  - 6.4 Million tunable parameters
  - Input dimension to the classifier is $|V_p|$ (the classifier takes a frequency vector).
  - The sequential model consists of a linear input layer ($a = W_1\mathrm{X} + b_1$) where W_1 is the first weights vector, X is the input vector, and b_1 is the first vector of biases, a ReLU activation function ($h = \max(0, a)$), where h is the hidden features vector, and another linear layer ($y = W_2 h + b_2$) to map the hidden features to output logits.

4.6 Clustering Methods

- Originally planned on researching and implementing a clustering method, but adjusting the n_neighbours parameter had a significant impact on the clustering, and further clustering methods were not necessary.

5 Assessment and Evaluation Tools

5.1 Clustering Validation

- Validation of clustering involved visual inspection and comparison to known phylogenetic trees.

5.2 Classification Accuracy

- Measured by the percentage of correctly classified language samples.

$$\text{Accuracy} = \frac{\text{Correct Guesses}}{\text{Total Guesses}}$$

- Precision, Recall, and F1 Score (harmonic mean of precision and recall) which is particularly useful for multi-class classification tasks.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

6 Results

6.1 Similarity Matrix

The first outcome from this project was a similarity matrix, where the rows and columns represent the languages within the dataset. The ij^{th} entry in this matrix represents the cosine similarity of the language corresponding to the i^{th} row and the j^{th} column. From left to right or top to bottom, our rows/columns are labelled as follows: Dutch, English, French, German, Indonesian, Italian, Portuguese, Romanian, Spanish, Swedish, Turkish, and Vietnamese.

```
[[1.  , 0.32, 0.41, 0.67, 0.29, 0.28, 0.31, 0.34, 0.42, 0.54, 0.41, 0.09],
 [0.32, 1.  , 0.41, 0.32, 0.22, 0.42, 0.34, 0.35, 0.33, 0.3 , 0.23, 0.22],
 [0.41, 0.41, 1.  , 0.38, 0.18, 0.54, 0.55, 0.51, 0.64, 0.36, 0.26, 0.05],
 [0.67, 0.32, 0.38, 1.  , 0.24, 0.33, 0.27, 0.28, 0.34, 0.48, 0.33, 0.09],
 [0.29, 0.22, 0.18, 0.24, 1.  , 0.26, 0.21, 0.22, 0.22, 0.28, 0.38, 0.22],
 [0.28, 0.42, 0.54, 0.33, 0.26, 1.  , 0.55, 0.57, 0.57, 0.33, 0.31, 0.08],
 [0.31, 0.34, 0.55, 0.27, 0.21, 0.55, 1.  , 0.5 , 0.79, 0.31, 0.26, 0.04],
 [0.34, 0.35, 0.51, 0.28, 0.22, 0.57, 0.5 , 1.  , 0.52, 0.3 , 0.31, 0.05],
 [0.42, 0.33, 0.64, 0.34, 0.22, 0.57, 0.79, 0.52, 1.  , 0.37, 0.29, 0.05],
 [0.54, 0.3 , 0.36, 0.48, 0.28, 0.33, 0.31, 0.3 , 0.37, 1.  , 0.36, 0.08],
 [0.41, 0.23, 0.26, 0.33, 0.38, 0.31, 0.26, 0.31, 0.29, 0.36, 1.  , 0.06],
 [0.09, 0.22, 0.05, 0.09, 0.22, 0.08, 0.04, 0.05, 0.05, 0.08, 0.06, 1.  ]]
```

Figure 3: Similarity Matrix based on Cosine Similarities

Examining the plot, we can observe that Dutch is most similar to German (0.67 similarity), French is very similar to Spanish (0.64 similarity), and Spanish is very similar to Portuguese (0.79 similarity), whereas French is very different from Indonesian (0.18 similarity). An interesting observation is that Vietnamese has incredibly low similarity with every other language in the dataset, which suggests that Vietnamese has a very unique orthographic structure. Looking at a small sample of text from Vietnamese, and comparing it to Spanish, this can be verified.

Vietnamese: Căn phòng im lặng ngay khi cánh cửa kêu cót két mở ra.
Spanish: La habitación quedó en silencio en cuanto la puerta chirrió al abrirse.

It is evident that Vietnamese commonly uses unique diacritics, which would create a language vector unlike any of the other languages.

6.2 Resulting plot from UMAP Dimensionality Reduction

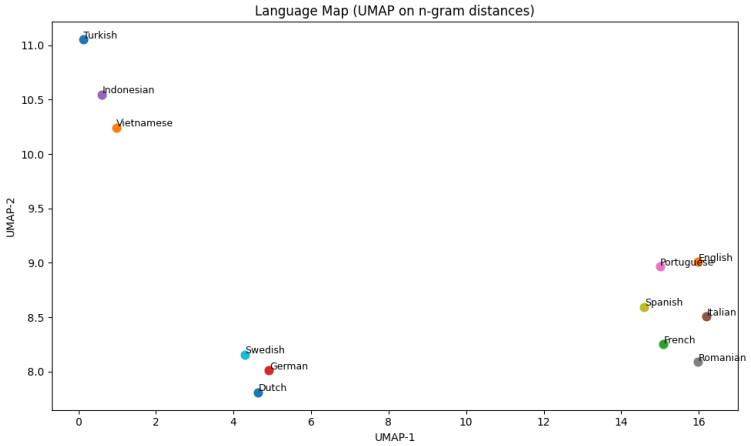

Figure 4: Final Plot after adding Turkish, Indonesian and Vietnamese to the dataset. For best results, n_neighbors = 3 was used.

Initially, the goal was to verify that languages would cluster with the rest of the languages in its respective language family. We can observe this effect with the Romance languages (Portuguese, Spanish, Italian, French, Romanian), as well as the Germanic languages (Swedish, German, Dutch), and languages that don't fall into either of these categories are clearly seperated from these clusters. It is also important to mention that distance between clusters is meaningless [2]. For cluster validation, silhouette score [5] was considered, but since the distance between clusters is irrelevant, this metric is not meaningful either.

The surprising result from this plot is the placement of the English sound profile vector. English has been grouped with the Romance languages, despite its classification as a Germanic language. Linguists believe that English shares a common ancestor with other Germanic languages, but has diverged phonetically due to historical contact with the Romance languages [6]. Due to this, English has developed a orthographic structure that resembles that of the Romance languages, hence why it was placed among this cluster.

6.3 Result of Cross-validation of the Classification Model

While Lightning Modules typically split data into training data and validation data to monitor the models ability to generalize information during training, we are not performing a validation step during training of this model. In this experiment, we are not tuning any hyper-parameters, and we are using a separate dataset to cross validate the model after training. Thus, a validation step during the training loop is unnecessary. If we split the data into 80% training data, and 20% validation data, the model is missing 20% of the data in each language's feature vector. With a vocabulary size of 49786 trigrams, this is almost 10000 missing datapoints. During testing, the validation step resulted in a substantial drop in classification accuracy. Our goal is to have the model learn the complete sound profile for each language, so it must be exposed to the full training set. Training on the full dataset avoids this problem, while still allowing generalization to be assessed during cross validation using the separate dataset of test sentences.

We have employed GPT-5.1 to generate 1113 random sentences varying in length, complexity and writing style. They are all labelled with their respective languages, and we are feeding this dataset into the model, performing cross-validation to assess the classification accuracy. We log the sentences that are incorrectly classified, and track various metrics needed to analyze the performance of the model. Below is a truncated output containing a sample of the incorrectly classified sentences, as well as the overall accuracy, number of sentences classified, incorrectly classified sentences, average length of incorrectly classified sentences, accuracy (only considering sentences with more than 3 words) and number of incorrectly classified sentences with more than 3 words.

```
--------------------------------------------------
Sentence:           Bra idé.
Actual language:    Swedish
Predicted Language: Spanish
--------------------------------------------------
Sentence:           Alles klar.
Actual language:    German
Predicted Language: Swedish
--------------------------------------------------
Sentence:           Solo vete.
Actual language:    Spanish
Predicted Language: Italian
--------------------------------------------------
Sentence:           Va juste.
Actual language:    French
Predicted Language: Romanian
------------------------------------------------------------
Accuracy: 94.52%
Total sentences classified: 1113
Wrongly classified sentences: 61
Average length of incorrectly classified sentences: 2.18 words.
Accuracy (only considering sentences with >3 words): 99.64%
Wrongly classified sentences (with >3 words): 3
```

We see that our overall accuracy is 94.52%. This is great, but we should explore to find out more about the shortcomings of our model. An interesting observation is that the average length of the incorrectly classified sentences is 2.3 words. For these short samples of text, the model simply doesn't have enough information to make a confident deduction. This is confirmed when we calculate the accuracy only considering the sentences containing more than 3 words. The accuracy for text samples containing more than 3 words is an astounding 99.64%.

6.4 Result of Scikit-learn's Classification report

Finally, from Scikit-learn, we are running a classification report on the model using the same test sentences dataset. We obtain a precision score,

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

a recall score,

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

and an F1-score for each language in the dataset, which is the harmonic mean of precision and recall.

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

These metrics tell us more about whether errors arise from overfitting, underfitting, or both [7]. These results show us that our model generalizes extremely well across our set of languages, and errors occur primarily between languages with similar orthographic text patterns, creating similar sound profiles. The near perfect scores for multiple languages tells us that the n-grams frequency vectors were an effective method of representing the sound profiles of languages, and our model was effective at generalizing information gathered from text samples to make a guess at the language.

	precision	recall	f1-score	support
Dutch	0.96	0.97	0.96	97
English	0.96	0.90	0.93	99
French	0.96	0.94	0.95	99
German	0.94	0.95	0.94	95
Indonesian	0.96	0.97	0.97	76
Italian	0.89	0.94	0.91	99
Portuguese	0.95	0.91	0.93	99
Romanian	0.96	0.95	0.95	99
Spanish	0.89	0.93	0.91	99
Swedish	0.97	0.93	0.95	99
Turkish	0.94	0.99	0.96	76
Vietnamese	1.00	1.00	1.00	76
accuracy			0.95	1113

```
   macro avg       0.95      0.95      0.95      1113
weighted avg       0.95      0.95      0.95      1113
```

7 Discussion

The results of this experiment confirm that trigram frequencies are effective features for language classification and create accurate numerical representations of orthographic structure. The similarity matrix reflected the expected relationships within language families - specifically Romance and Germanic. These patterns were shown once again in the plot using the UMAP embedding, which produced distinct clusters.

During evaluation of the MLP, the main weakness involved very short input samples. With too little trigram frequency data, the model simply did not have enough information and often "picked" languages with high similarity in orthographic structure.

Regardless, the classifier performed exceptionally well across all languages, achieving a weighted F1 score of approximately 0.95. This indicates that the model was sufficiently sized to "learn" orthographic patterns contained in the trigram frequency feature vectors and generalize them effectively.

In the end, the quantitative results (accuracy, precision, recall, F1) and qualitative results (similarity matrix and UMAP visualization) validate the approach of using character level n-grams to capture meaningful linguistic data, and that a simple neural network can effectively distinguish languages using this representation.

8 Conclusion

This project developed a complete workflow for representing and analyzing languages through character level trigram frequencies. The approach produced meaningful measures of similarity between languages and generated a clear two dimensional embedding that reflected known linguistic groupings. The classification component demonstrated that a simple MLP can effectively use these representations, achieving high accuracy and strong evaluation metrics across all languages.

The project was a success in meeting its objectives to validate the use of character level n-grams to create a numerical representation of orthographic structure, and produce interpretable visualizations of the local relationships between languages as well as performing language classification using a simple MLP using this representation.

9 Future Work

Several improvements could be added to produce new results, aswell as enhance the accuracy of the model and the effectiveness of the n-gram representation of the languages.

- Including phonetic information:
  - Using IPA based representations of text would enable comparisons of languages outside of those that use latin script, and would greatly improve the phonetic representation of each language. This would also improve the accuracy of the classifier, as accurate phonetic representations would more closely represent the sound and structure of languages, as well as increase the vocabulary size, causing the representation to be more unique across each language.
  - Using a faster Grapheme-to-Phoneme (G2P) model or exploring other methods of collecting IPA text could allow this improvement
- Developing an interactive visualization tool:
  - Initially, the plan for this project was to create an interactive map to explore relationships between languages.
  - In the future, integrating the results found and methodologies used in this exploratory project into a web app or desktop tool could be a natural extension, allowing users to visualize language similarity, inspect UMAP embeddings, and run language classification on sample text.

10 References

References

[1] O. Hankare, "Exploring n-grams: The building blocks of natural language understanding," Mar 2024, https://ompramod.medium.com/exploring-n-grams-the-building-blocks-of-natural-language-understanding-d40a7a309d12.

[2] A. P. Andy Coenen, "Understanding umap," 2025, https://pair-code.github.io/understanding-umap/.

[3] lingjzhu, "Github - lingjzhu/charsiug2p: Multilingual g2p in 100 languages," 2022, https://github.com/lingjzhu/CharsiuG2P.

[4] U. L. Documentation, "Umap basic parameters," 2018, https://umap-learn.readthedocs.io/en/latest/parameters.html.

[5] S. L. Documentation, "Silhouette score," 2025, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html.

[6] M. Filppula and J. Klemola, "External influences in the history of english," *Oxford Research Encyclopedia of Linguistics*, Aug 2020. [Online]. Available: https://oxfordre.com/linguistics/display/10.1093/acrefore/9780199384655.001.0001/acrefore-9780199384655-e-284?p=emailAamU6LYPfIEtg&d=/10.1093/acrefore/9780199384655.001.0001/acrefore-9780199384655-e-284

[7] btd, "Model fitness check: 11 evaluation metrics to identify overfitting and underfitting in ml models," Nov 2023. [Online]. Available: https://baotramduong.medium.com/machine-learning-overfitting-vs-underfitting-96fe8b41192b

[8] W. Cavnar and J. Trenkle, "N-gram-based text categorization," https://dsacl3-2019.github.io/materials/CavnarTrenkle.pdf.

[9] S. L. Documentation, "2.3 clustering," 2019, https://scikit-learn.org/stable/modules/clustering.html#k-means.

[10] S. Yıldırım and T. Yıldız, "A comparative analysis of text classification for turkish language," *ResearchGate*, vol. 24, no. 5, 2018, https://www.kaggle.com/savasy/ttc4900.

[11] T. Nguyen, "Vietnamese text classification dataset," 2022. [Online]. Available: https://www.kaggle.com/datasets/tuannguyenvananh/vietnamese-text-classification-dataset

[12] Google. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall

[13] N. Sharma, June 2023. [Online]. Available: https://arize.com/blog-course/f1-score/