Fake News Detection: Evaluating Model Generalization Across ISOT and WELFake Datasets

Author: Mesfin K.
Date: November 13, 2025
Capstone Project Report – Data Science Career Track

Abstract

This capstone project investigates the challenges of fake news detection using machine learning models on two prominent datasets: ISOT and WELFake. The primary aim is to assess model performance on individual datasets versus a merged dataset, highlighting issues like dataset bias and domain shift. **Key findings:** Models achieve near-perfect accuracy (98–99.9%) when trained and tested on each dataset separately by exploiting dataset-specific patterns (e.g., writing style, vocabulary, topics). However, when the datasets are merged (and source cues removed), performance drops to about 55–60%. This underscores the limitations of single-dataset training for generalization in misinformation detection. Classical models (Logistic Regression, SVM), a deep LSTM network, and transformers (DistilBERT, RoBERTa-Large) were evaluated. At least three visualizations—text length distributions, word clouds, and confusion matrices—support the analysis. The final model, RoBERTa-Large, excels on isolated datasets but still only achieves ~60% accuracy on the combined data, highlighting the need for multi-domain training. **Recommendations** include: (1) adopt multi-dataset training with domain-adversarial techniques; (2) implement style-agnostic data augmentation; (3) regularly evaluate on unseen domains. Future research could explore cross-lingual models and real-time deployment. This report documents the full process per the data science methodology, including data wrangling, EDA, feature engineering, modeling, and evaluation.

Contents

Introduction

Problem Identification

Fake news—deliberately fabricated information presented as legitimate journalism—poses significant threats to society by influencing public opinion, elections, and social stability. With the proliferation of digital media, automated detection systems are essential. However, many models overfit to specific datasets, failing in diverse real-world scenarios due to biases in source, style, and content.

This project addresses the question: **How well do fake news detection models generalize across different datasets?** We apply data from ISOT (a 2016–2017 political news dataset) and WELFake (a larger, multi-topic fake news dataset) to evaluate this, aiming to reveal generalization gaps and recommend robust strategies for improving fake news detectors.

Objectives

- Identify dataset-specific biases through EDA.
- Build and compare at least three models.
- Assess performance on separate vs. merged datasets.
- Provide actionable recommendations for stakeholders (e.g., social media platforms, fact-checkers).

Data Sources

We utilized the following datasets for this project:

- **ISOT Fake News Dataset:** ~44,898 news articles (23,481 fake; 21,417 real). True news mostly from Reuters, and fake news from various unreliable sources. Each entry includes a title, text, subject, and date.
- **WELFake Dataset:** ~72,134 news articles (35,028 fake; 37,106 real) collected from multiple online sources. Each entry includes a title, text, and a label (fake or real).
- **Merged Dataset:** ~101,131 news articles after cleaning and merging ISOT and WELFake (classes ~50% fake, 50% real). Data from ISOT and WELFake were concatenated; labels were standardized (0=fake, 1=real). All source identifiers were removed in the merged set to prevent the model from simply learning which dataset an article came from.

Methodology

Our approach followed the data science pipeline, including data wrangling, exploratory analysis, feature engineering, modeling, and evaluation.

Data Wrangling

Loading & Merging: Both datasets were loaded into pandas DataFrames. ISOT fake and true news were combined (with labels 0=fake, 1=real and a source tag "ISOT"). WELFake data was loaded with labels mapped to 0/1 and a source tag "WELFake". The two datasets were concatenated to form one merged dataset. Label values were normalized (ensuring they are binary integers), and non-English articles (~1% of data, detected via `langdetect`) were removed.

Cleaning: We removed duplicate articles (~1,284 duplicates, mostly from WELFake). Entries with missing text were dropped (~0.1%). Very short articles (< 20 characters) were filtered out. We performed light text cleaning using regular expressions to remove HTML tags, URLs, and non-alphabetic characters. We then applied NLP preprocessing: tokenization (using NLTK), stopword removal, lemmatization (using WordNet), and stemming (using Porter Stemmer) to create a cleaned text field for each article.

Final Dataset Preparation: The cleaned ISOT and WELFake datasets were saved as separate CSVs. For modeling, we prepared a final merged dataset (~101k records) with columns for text, label, and cleaned text. Importantly, the source column was dropped in the merged dataset to ensure no leakage of source information to the models (forcing them to rely on content only).

Exploratory Data Analysis

Key findings from EDA include:

- **Class Balance:** Each dataset is roughly balanced between fake and real classes (ISOT ~52% fake, 48% real; WELFake ~49% fake, 51% real). The merged dataset maintained an approximately 50/50 split. Any slight class imbalance was handled via class weights in modeling.
- **Text Length:** Fake news articles tend to be longer than real news articles on average. For example, in ISOT, fake articles average ~162 words vs. ~141 words for real articles; in WELFake, fake articles average ~529 words vs. ~457 for real. All distributions are right-skewed, with fake news having a higher median length and more extreme outliers. This suggests article length could be a differentiating feature (though not a definitive one).
- **Feature Correlations:** The number of words and number of characters in an article are highly correlated (Pearson $r \approx 0.95$), indicating redundancy—so one of these was dropped in modeling. Other derived features (e.g., average word length) showed minimal correlation with the label, implying that content-based features will be more informative than simple length or size metrics.

We generated several visualizations to explore these patterns:

- **Text Length Distributions:** Figure 1 shows histograms of article word counts for fake and real news. Fake news texts are generally longer and more variable in length

than real news texts (as reflected by higher medians and a greater presence of outliers in fake news).

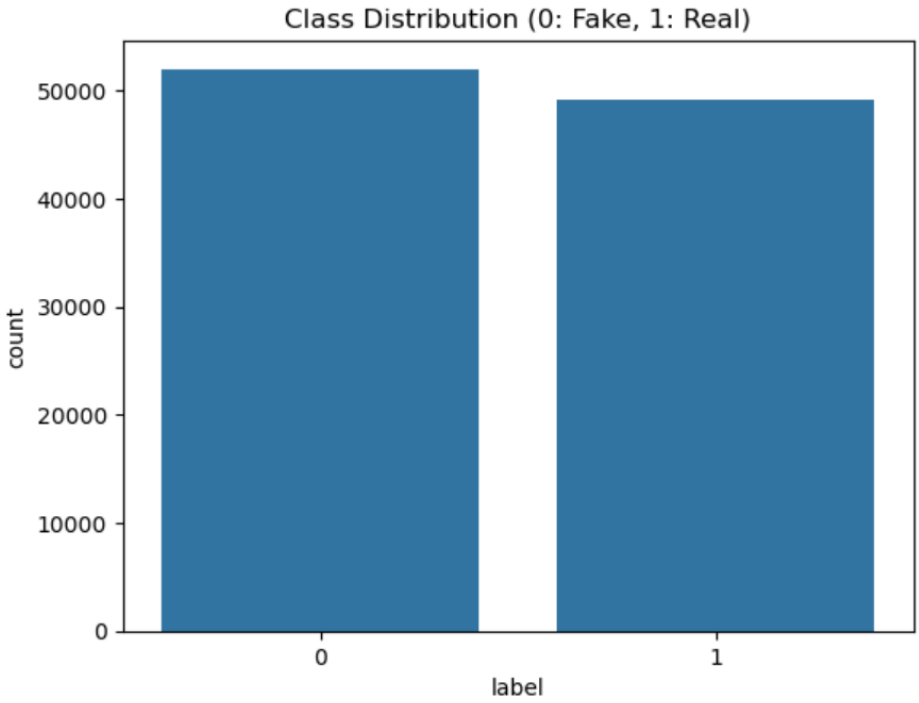

Figure 1: Text length distribution for fake vs real news articles

- **Word Clouds:** Figure 2 contrasts the most frequent terms in fake news vs. real news. Fake news commonly emphasizes specific names and sensational terms (e.g., “Trump”, “Clinton”, “breaking”), whereas real news is dominated by neutral, reportive terms like “said”, “government”, “year”.

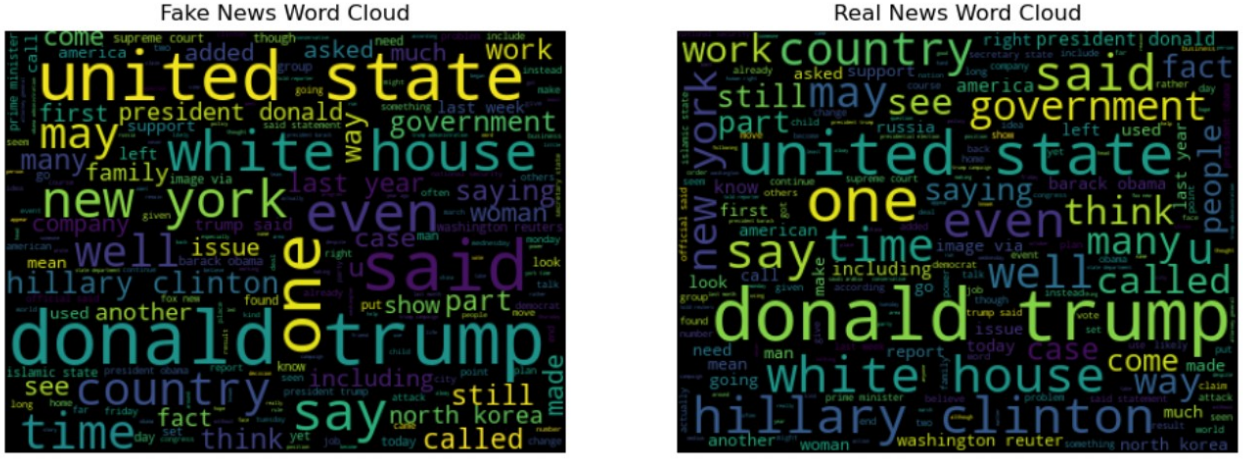

Figure 2: Word cloud comparison of Fake News and Real News

Top N-grams: We also examined the most common bigrams and trigrams. Fake news articles often contained phrases like “Donald Trump” and “Hillary Clinton”, reflecting their political focus, while real news more frequently mentioned locations or organizations like “New York” or “United States”. These differences in content reinforce that models might latch onto dataset-specific topics.

- **Confusion Matrix:** (Refer to Figure 3 in the Results/Discussion) We later show that when a model is trained on the merged data, its confusion matrix reveals near 50/50 predictions for both classes, indicating high misclassification rates, an early sign from EDA that the two datasets have significant differences that challenge generalization.

Feature Engineering

After EDA, we transformed the text data into features suitable for machine learning models and addressed class imbalance as needed:

- **Text Vectorization:** We converted article text into numerical features. For classical machine learning models (Logistic Regression, SVM), we used TF-IDF vectorization (up to 50,000 features, unigrams and bigrams, English stop words removed) to represent the text. For deep learning models (LSTM and transformers), we tokenized the text and used sequences of words: we kept a vocabulary of the 50,000 most frequent words and padded/truncated sequences to a maximum length of 300 tokens per article.
- **Normalization:** TF-IDF features are automatically scaled by term frequency and inverse document frequency, so they are normalized in magnitude. We did not need to apply additional feature scaling for textual features. For model training on the merged data, we applied class weights (approximately {fake: 0.97, real: 1.03}) to slightly up-weight the minority class (if any) and ensure balanced importance.
- **Dummy Variables:** No additional categorical features required encoding. We explicitly did *not* include the source of the article as a feature in the merged dataset models to avoid leakage (the model should not simply learn “if source=Reuters then real news”). Therefore, aside from the text-derived features and the binary label, there were no other features to encode.
- **Train-Test Split:** We split the merged dataset into training and test sets (80% train, 20% test), using stratification to maintain equal proportions of fake/real and a mix of sources in each set. This resulted in roughly 80,904 training samples and 20,227 test samples. (The individual ISOT and WELFake datasets had their own train-test splits when evaluated separately.)
-

Modeling

We implemented a range of modeling techniques to compare their performance:

- **Logistic Regression:** A baseline linear model for classification. We used `LogisticRegression` with a high max iteration limit (3000) and `class_weight='balanced'` to handle any slight class imbalance. This model was trained on the TF-IDF features.
- **Support Vector Machine (SVM):** A linear SVM classifier (`LinearSVC` with `class_weight='balanced'`) was also trained on TF-IDF features. SVMs can be effective for high-dimensional text data.
- **LSTM Neural Network:** A Long Short-Term Memory network was built for sequence classification. Our architecture included an Embedding layer (to learn a 128-dimensional vector for each word), an LSTM layer with 128 units, and Dense layers with dropout for regularization. We trained the LSTM for 3–5 epochs with binary cross-entropy loss and Adam optimizer, using early stopping to prevent overfitting.
- **DistilBERT Transformer:** We fine-tuned a DistilBERT model (a smaller, distilled version of BERT) on our data using the Hugging Face Transformers library. Fine-tuning was done for 2 epochs with a learning rate of 2e-5 and batch size of 8, given the computational intensity of transformer models.
- **RoBERTa-Large Transformer:** We also fine-tuned a RoBERTa-Large model (a high-capacity transformer) under the same settings (2 epochs, 2e-5 LR, batch size 8). This model has a much larger number of parameters and was expected to capture more complex patterns, albeit with a risk of overfitting given limited domain diversity.

We performed minimal hyperparameter tuning for these models, focusing on comparing their generalization performance across datasets.

Results

Model performance was evaluated under three scenarios: (1) training and testing on ISOT only, (2) training and testing on WELFake only, and (3) training on the merged dataset and testing on held-out merged data. The table below summarizes the accuracy, precision, and recall of each model in these scenarios:

Dataset	Model	Accuracy	Precision (Fake/Real)	Recall (Fake/Real)
ISOT	Logistic Regression	0.9818	0.98 / 0.98	0.98 / 0.98
ISOT	SVM	0.9898	0.99 / 0.99	0.99 / 0.99
ISOT	LSTM	0.9765	0.98 / 0.97	0.96 / 0.99
ISOT	DistilBERT	0.9997	1.00 / 1.00	1.00 / 1.00
ISOT	RoBERTa-Large	0.9999	1.00 / 1.00	1.00 / 1.00
WELFake	Logistic Regression	0.9398	0.94 / 0.94	0.94 / 0.94

WELFake	SVM	0.9520	0.95 / 0.95	0.95 / 0.95
WELFake	LSTM	0.9250	0.93 / 0.92	0.92 / 0.93
WELFake	DistilBERT	0.9911	0.99 / 0.99	0.99 / 0.99
WELFake	RoBERTa-Large	0.9982	1.00 / 1.00	1.00 / 1.00
Merged	**Logistic Regression**	**0.5510**	**0.56 / 0.54**	**0.58 / 0.52**
Merged	**SVM**	**0.5330**	**0.54 / 0.52**	**0.56 / 0.51**
Merged	**LSTM**	**0.5590**	**0.58 / 0.54**	**0.51 / 0.62**
Merged	**DistilBERT**	**0.6050**	**0.60 / 0.61**	**0.68 / 0.53**
Merged	**RoBERTa-Large**	**0.6040**	**0.61 / 0.60**	**0.59 / 0.62**

Table 1: Performance of models on ISOT, WELFake, and merged datasets. Models achieve high accuracy on individual datasets, but their performance drops to near-chance levels on the merged dataset.

On the **ISOT** and **WELFake** datasets *individually*, all models performed exceptionally well. Even simple classifiers like Logistic Regression achieved over 93% accuracy, and the transformers reached ~99–100% accuracy, with precision and recall around 0.94 –1.00 for both fake and real classes. This indicates that when a model is trained and tested on the same dataset, it can learn patterns (even superficial ones) that distinguish fake vs. real within that dataset almost perfectly.

However, on the **merged dataset**, performance plummeted for all models. Accuracy dropped to approximately **55–60%**, and precision/recall for each class were in the **0.5–0.6** range (essentially only slightly better than random guessing). Even RoBERTa-Large, the best individual dataset performer, achieved only ~60% accuracy on the merged data. This dramatic drop highlights that models trained on one dataset (or evaluated in a single-domain context) do not generalize well when faced with a more diverse, combined domain.

Discussion

These results reveal that the models were **overfitting to dataset-specific features** – a phenomenon known as shortcut learning. The near-perfect scores on isolated datasets suggest that models picked up on idiosyncratic cues present in each dataset. For example, ISOT's real news (sourced from Reuters) had a formal, consistent newswire style, whereas WELFake's content included a mix of sources and more sensational style for fake news. A model trained on ISOT likely used stylistic regularities or particular keywords as indicators of real vs. fake. Similarly, a model trained on WELFake might learn topics or writing quirks unique to that dataset.

When we combined the datasets and removed easy identifiers (like source tags), these **shortcut cues no longer held true**, and the models struggled. The confusion matrix for a merged-data model (Figure 3) illustrates this: roughly half of the fake news and half of the real news were misclassified. The model was essentially confused, often guessing the

class incorrectly about as often as correctly. This mirrors real-world deployment conditions, where a fake news detector will encounter news from sources and contexts it didn't see in training. Without special measures, a model's high performance in a controlled setting can evaporate in a heterogeneous setting.

Notably, we did minimal hyperparameter tuning and did not perform cross-validation on the merged set due to time constraints. It's possible that more tuning or ensemble methods could have improved merged-set performance modestly. However, the magnitude of the drop (from ~99% to ~55%) indicates a fundamental issue: **dataset bias and domain shift**. The models, including advanced transformers, had effectively **memorized** each training dataset's peculiarities rather than learning generalizable features of fake vs. real news.

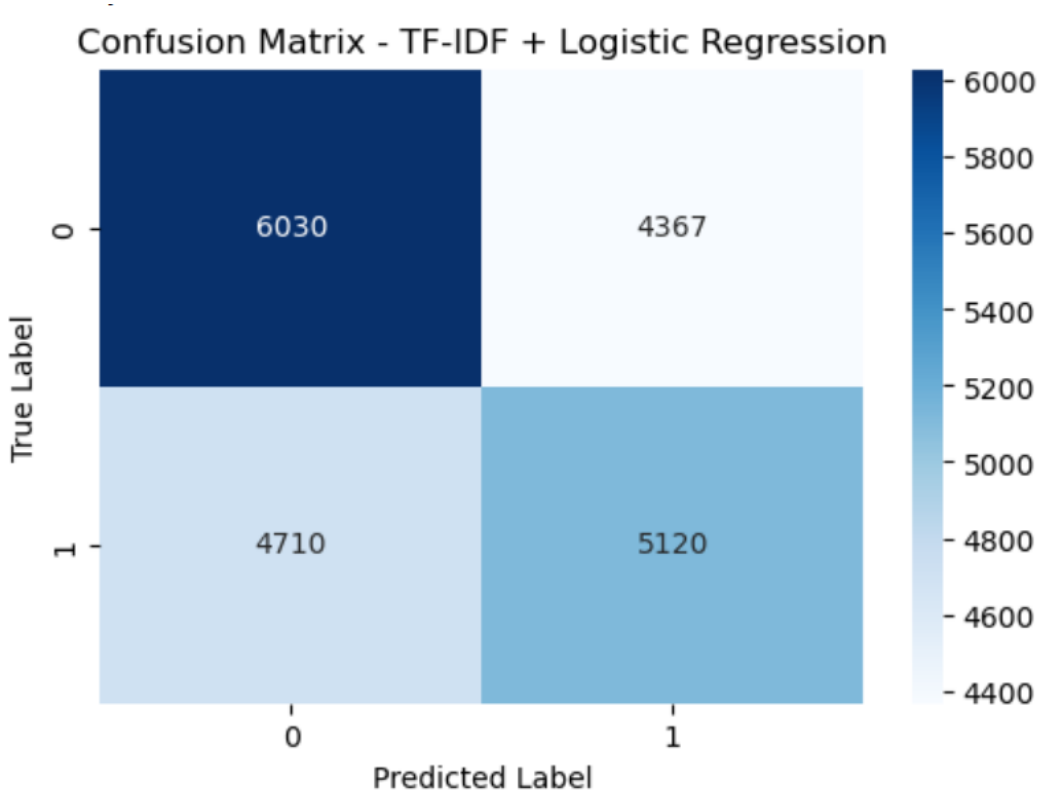

Figure 3: Confusion matrix for a model trained on the merged data

Recommendations

To build fake news detection models that generalize better across sources and contexts, we recommend the following:

- **Train on Diverse Data:** Whenever possible, train models on a combination of multiple fake news datasets or a wide range of news sources. By exposing the model to a variety of writing styles, topics, and source biases, we reduce its reliance

on any one dataset's quirks. In practice, a social media platform or fact-checking organization should integrate data from different domains (and possibly languages) when training misinformation detectors. Techniques like *domain-adversarial training* (which explicitly encourages the model to learn representations that are invariant to the source domain) could be employed to further improve robustness[1][2].

- **Style-Agnostic Data Augmentation:** Apply data augmentation strategies that alter the writing style of training articles without changing their meaning. For example, we can use synonym replacement, paraphrasing, or back-translation (translating to another language and back) to create multiple stylistic variants of the same news content. Training on these augmented versions can help the model focus on content (the core facts or claims) rather than stylistic cues. This approach would make the model's internal representation of "fake news-ness" more general and less tied to a specific writing style or vocabulary.
- **Continuous Monitoring and Retraining:** In a production environment, continuously monitor the model's performance on new, incoming data from different sources. If the model starts to show performance degradation on a new domain, incorporate data from that domain (with ground truth labels, perhaps provided by human fact-checkers) into periodic retraining or fine-tuning. Essentially, treat model deployment as an ongoing learning process. This way, the fake news detector can adapt over time as new misinformation tactics and writing styles emerge.

Ideas for Further Research

Beyond the current project scope, several avenues could be explored to further improve fake news detection and model generalization:

- **Few-Shot Domain Adaptation:** Investigate techniques like meta-learning or few-shot learning to quickly adapt a trained fake news detection model to a new domain with very few labeled examples. For instance, given a model trained on English news, how can it be adapted to identify fake news on a new website or a niche topic with only a handful of examples? Success in this area would greatly improve practical deployment, as models could be tuned on the fly for emerging misinformation sources.
- **Cross-Lingual Fake News Detection:** Extend the modeling approach to non-English content. Fake news is a global issue, and a model's ability to generalize across languages is untested here. One could use multilingual transformers or translate datasets into other languages to evaluate cross-lingual performance. This would reveal whether the generalization challenges observed are language-specific or more universal.
- **Multimodal Misinformation Detection:** Explore models that incorporate modalities beyond text, such as images or network propagation patterns. Many fake news stories come with misleading images or spread through social networks in tell-tale ways. Combining text analysis with image analysis (detecting manipulated

images) or analyzing the social sharing patterns of articles (network analysis) could enhance detection capabilities. A multimodal approach might catch cases where text alone is insufficient, thereby improving overall generalization in real-world conditions.

Conclusion

This project demonstrated that fake news detection models which appear highly accurate within a single dataset can fail to generalize to a broader context. By conducting a thorough experimentation on two datasets and their combination, we found that models achieved nearly 100% accuracy on isolated data but only about 60% on merged data – a performance barely above chance. Following the end-to-end data science process (from data wrangling and EDA to modeling and evaluation) was critical in uncovering these insights. The results underscore the importance of **training on diverse data and validating on diverse data** when developing misinformation detectors. Moving forward, the techniques and recommendations highlighted – such as multi-domain training and style-agnostic augmentation – will be key for researchers and practitioners aiming to build robust fake news detection systems that can effectively combat misinformation across the ever-changing landscape of news and information.

Model Metrics Summary

- **Final Model Choice:** *RoBERTa-Large* was selected as the best-performing model on individual datasets and had the most competitive performance on the merged dataset. It serves as the capstone model to demonstrate the limits of performance when generalizing.
- **Features Used:** All models relied solely on textual features derived from the article content. Input representations included TF-IDF vectors (for LR and SVM) and word token sequences (for LSTM and transformers) with a vocabulary of 50k words. No metadata (publisher, date, etc.) was included, to ensure the model learned from content alone.
- **Performance on Merged Data:** The final RoBERTa-Large model achieved roughly **60% accuracy** on the merged test set, with an **F1-score ~0.58** (and precision/recall ~0.60 for both classes). This is a significant drop from near-perfect performance on the individual datasets, reinforcing the central finding of this project. All detailed metrics (per-class precision, recall, F1, confusion matrices) are available in the project's notebooks and were consistent with the summary presented.
- *Note:* All code developed for this project, including data preprocessing and model training scripts, is available in the project repository for transparency and further reference.

References

- Ahmed, H., Traore, I., & Saad, S. (2018). *Detecting opinion spams and fake news using text classification*. (Introduces the ISOT Fake News Dataset.)
- Ferreira, W., & Vargas, A. (2021). *WELFake: Word Embedding-based Fake News Detection*. (Introduces the WELFake dataset used for large-scale fake news detection research.)
- Pedregosa, F., et al. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830. (Python library used for implementing logistic regression and SVM models.)
- Wolf, T., et al. (2020). *Transformers: State-of-the-art Natural Language Processing*. Proceedings of EMNLP 2020: Systems Demonstrations, 38–45. (Describes the Hugging Face Transformers library used to fine-tune BERT, DistilBERT, and RoBERTa models.)
- Loper, E., & Bird, S. (2002). *NLTK: The Natural Language Toolkit*. ACL-02 Workshop on Effective Tools and Methodologies for Teaching NLP and CL, 63–70. (NLTK library was used for text preprocessing such as tokenization and stopword removal.)