# Project Group 1

## Project Supervisor

**Prof. Uttam Kumar Roy**

## Project Group Members

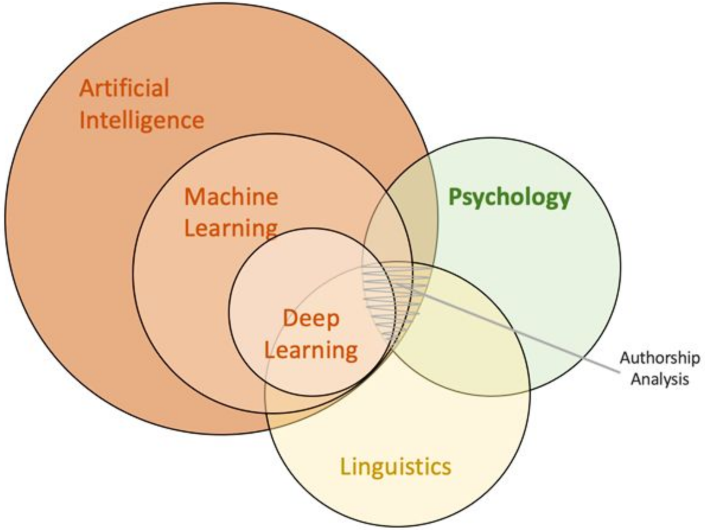
**Sumanta Kumar Paul**
(Roll No. - 002011001063)

**Sayan Maji**
(Roll No. - 002011001079)

**Soumyadeep Pal**
(Roll No. - 002011001113)

# Index

- ➢ Problem Statement
- ➢ Introduction
- ➢ Significance
- ➢ Linguistic Analysis
- ➢ Datasets Used
- ➢ Existing Approaches
- ➢ Proposed Approach
- ➢ GloVe Word Embeddings
- ➢ Multi-channel CNN Model
- ➢ Implementation
- ➢ Results and Analysis
- ➢ Scope for Future Work
- ➢ Conclusion

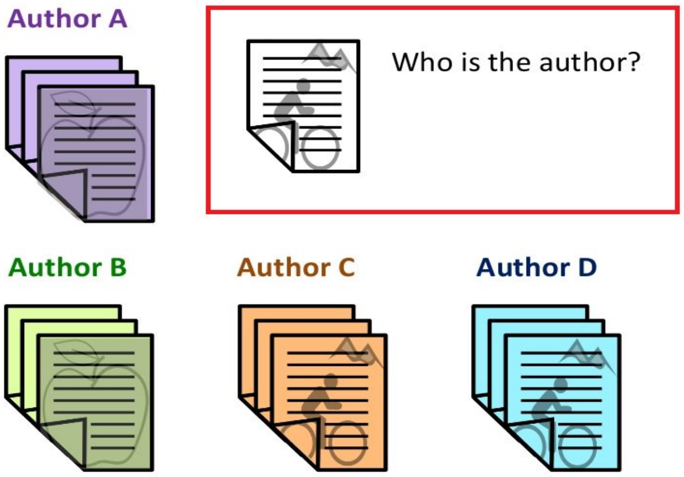


# Problem Statement

**Propose a machine learning model, with implementation of the approach, for authorship attribution of articles using linguistic analysis.**

# Introduction

**Authorship attribution** is the task of identifying the author of a text document written by an unidentified author, given a set of text documents written by a set of authors which includes the unidentified author.



**Project Objective -** This project attempts to perform authorship attribution on news and blog datasets using multi-channel convolutional neural networks (CNNs) with word embeddings and compare its performance with traditional machine learning methods using stylometric and other linguistic feature sets.

# Significance

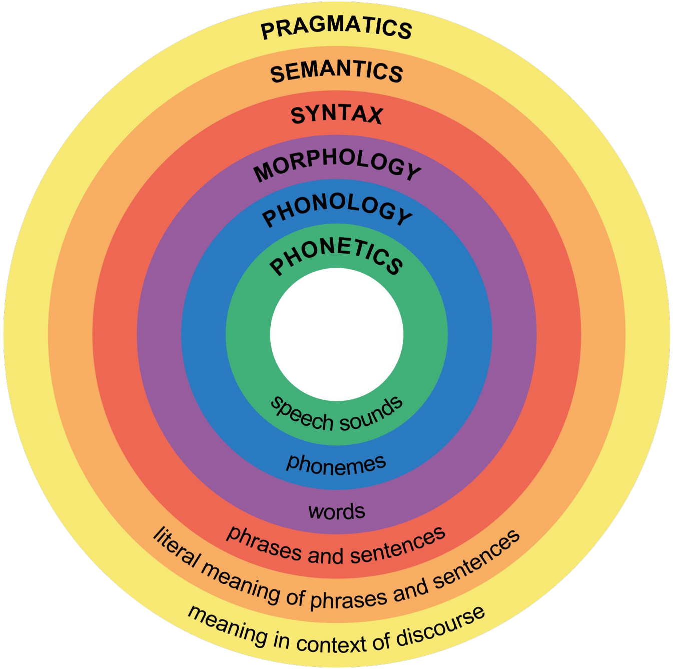
- **Law Enforcement -** Narrowing down list of suspects in cases of anonymous threat letters, ransom notes and hoax messages.
- **Plagiarism Detection -** Detection of plagiarism based on linguistic analysis for large text files where searching is inefficient.
- **Historical Knowledge -** Identification or confirmation of authors of old manuscripts and unfinished literary works with unknown authors.
- **Intellectual Property Protection -** Identification of real authors of news articles, research papers and literary works in case of disputes.
- **Fake News and Reviews Detection -** Detection of blogs and articles with provocative or defamatory content and fake or bot reviews.

# Linguistic Analysis

Linguistic analysis is the **systematic study of language** to understand its structure, meaning, and use. It involves breaking down language into its fundamental components to explore how it functions and conveys meaning.

Linguistic analysis can be performed in **three ways**:

1. Manual Comparison
2. Mathematical Formulae
3. Machine Learning Algorithms



# Linguistic Analysis: Techniques

- **Lexical analysis** involves the study and processing of words in a text, and is fundamental for spell checking, keyword extraction, and SEO.
- **Syntactic analysis** examines the grammatical structure of sentences and identifies the syntactic relationships between words and phrases.
- **Semantic analysis** focuses on understanding the meaning of words, phrases, and sentences and is essential for machine translation.
- **Structural analysis** examines the overall organization and arrangement of text and analyses the use of structural elements.
- **Pragmatic analysis** goes beyond literal meaning to consider the context of words, the speaker's intent and the situational context.
- **Sentiment analysis** determines the emotional tone of a text and is widely used in social media monitoring and feedback analysis.

# Linguistic Analysis: Techniques

- **Discourse analysis** studies large units of text such as conversations, paragraphs or documents, to understand how they convey meaning.
- **Morphological analysis** studies the structure of words and involves identifying root words, prefixes, suffixes, and inflections.
- **Idiosyncratic analysis** examines unique language use and individual variations in language like word choices, styles and expressions.
- **Stylometric analysis** examines linguistic features such as word frequency and sentence length to identify unique writing styles.
- **Sociolinguistic analysis** examines how language varies across different social groups and other demographic contexts like age, gender, ethnicity, etc.

# Linguistic Analysis: Feature Sets

- **Count Vectorization -** Involves tokenization of text documents into a matrix of token counts and comparing frequency of words.
- **TF-IDF Vectorization -** Represents importance of words by comparing for Term Frequency (as a proportion of total word count) and Inverse Document Frequency (proportion of documents containing the word).
- **Write-print Stylometry -** Involves lexical features (word counts, word lengths, bigrams and trigrams, vocabulary richness), syntactic features (function words and punctuations), structural features (sentence and paragraph arrangements), content-specific features (acronyms, keywords and jargons) and idiosyncratic features (common mistakes).

# Linguistic Analysis: Feature Sets

- **Basic-9 Feature Set -** Includes character count, unique words count, lexical density, average syllables per word, sentence count, average sentence length and some readability metrics.
- **Word Embeddings -** Based on co-occurrence of words and uses a combination of GloVe (Global Vector) embeddings and word-word embeddings to represent documents.

# Datasets Used

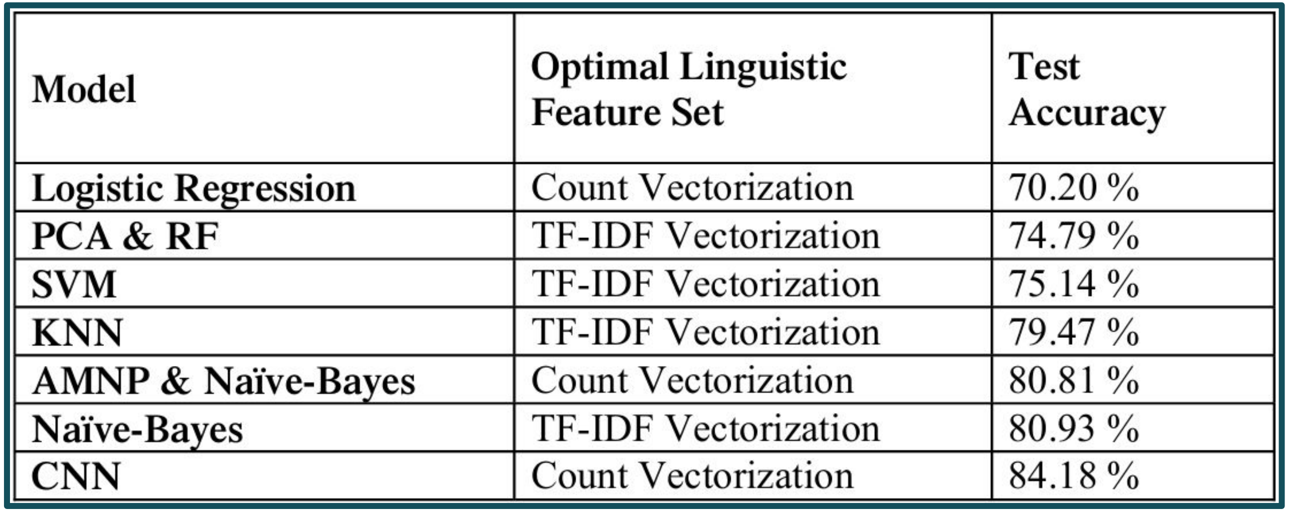
**The Reuters Corpus -** https://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

The Reuters Corpus is a collection of documents that appeared on Reuters newswire in 1987. The documents were assembled and indexed with categories. The original corpus has 10,369 documents and a vocabulary of 29,930 words.

**The Blog Authorship Corpus -** https://u.cs.biu.ac.il/~koppel/BlogCorpus.htm

The Blog Authorship Corpus consists of the collected posts of 19,320 bloggers gathered from blogger.com in August 2004. The corpus incorporates a total of 681,288 posts and over 140 million words - or approximately 35 posts and 7250 words per person.

# Existing Approaches



| Model | Optimal Linguistic Feature Set | Test Accuracy |
|---|---|---|
| Logistic Regression | Count Vectorization | 70.20 % |
| PCA & RF | TF-IDF Vectorization | 74.79 % |
| SVM | TF-IDF Vectorization | 75.14 % |
| KNN | TF-IDF Vectorization | 79.47 % |
| AMNP & Naïve-Bayes | Count Vectorization | 80.81 % |
| Naïve-Bayes | TF-IDF Vectorization | 80.93 % |
| CNN | Count Vectorization | 84.18 % |

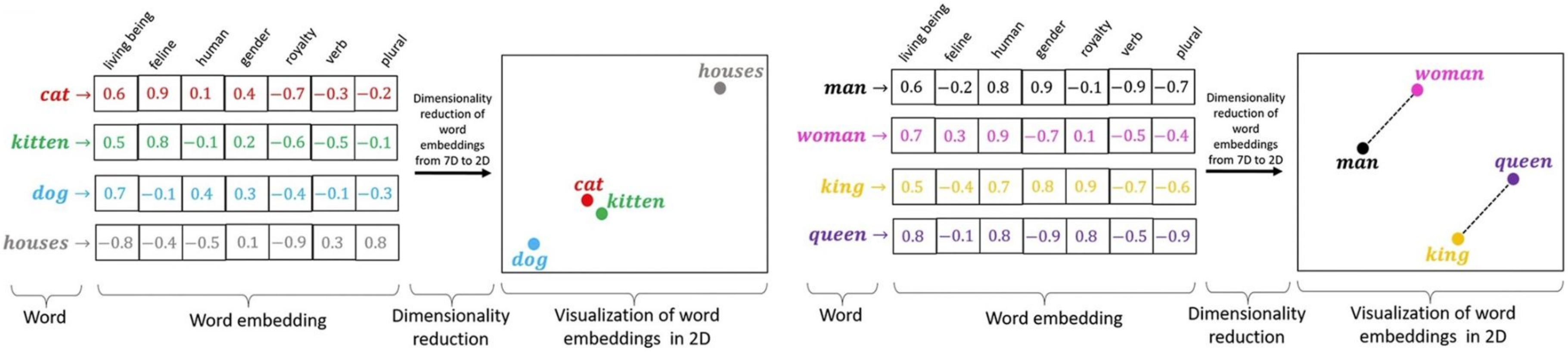
**Observation -** CNN yields the highest accuracy, hence, it is justifiable to try and improve it to obtain higher accuracy values.

# Proposed Approach

- The proposed approach uses **Convolutional Neural Network (CNN) classifier with word embeddings** for authorship attribution. Each word is mapped to a continuous-valued word vector using GloVe embeddings. Each input document is represented as a concatenation of word embeddings.

- The CNN model is trained using these document representations as input for authorship attribution. In other words, the **multi-channel CNN** consisting of a **static word embedding channel** (word vectors trained by GloVe embeddings) and a **non-static word embedding channel** (word vectors trained initially by GloVe embeddings then updated during training) is trained.

# GloVe Word Embeddings



| | living being | feline | human | gender | royalty | verb | plural |
|---|---|---|---|---|---|---|---|
| cat → | 0.6 | 0.9 | 0.1 | 0.4 | −0.7 | −0.3 | −0.2 |
| kitten → | 0.5 | 0.8 | −0.1 | 0.2 | −0.6 | −0.5 | −0.1 |
| dog → | 0.7 | −0.1 | 0.4 | 0.3 | −0.4 | −0.1 | −0.3 |
| houses → | −0.8 | −0.4 | −0.5 | 0.1 | −0.9 | 0.3 | 0.8 |

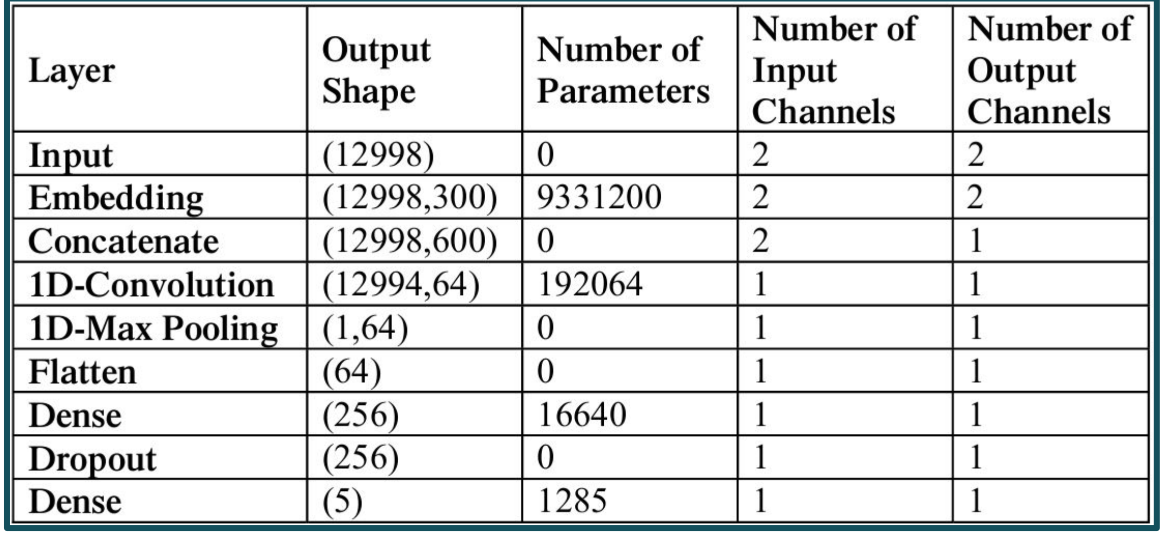
| | living being | feline | human | gender | royalty | verb | plural |
|---|---|---|---|---|---|---|---|
| man → | 0.6 | −0.2 | 0.8 | 0.9 | −0.1 | −0.9 | −0.7 |
| woman → | 0.7 | 0.3 | 0.9 | −0.7 | 0.1 | −0.5 | −0.4 |
| king → | 0.5 | −0.4 | 0.7 | 0.8 | 0.9 | −0.7 | −0.6 |
| queen → | 0.8 | −0.1 | 0.8 | −0.9 | 0.8 | −0.5 | −0.9 |

Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D
Dimensionality reduction of word embeddings from 7D to 2D
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D

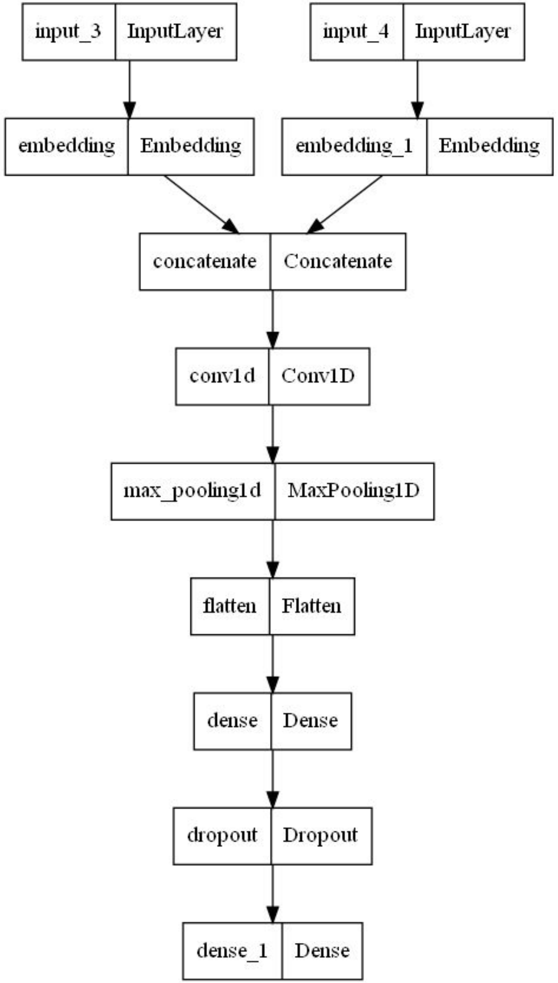
**GloVe (Global Vectors for Word Representation)** is a word embedding technique that combines both local context and global statistical information of a corpus to create word vectors. These embeddings are dense vector representations that capture semantic relationships between words by analyzing word co-occurrence statistics from a large corpus. Unlike traditional count-based models that rely solely on local context information, GloVe constructs a global word-word co-occurrence matrix.

# Multi-channel CNN Model

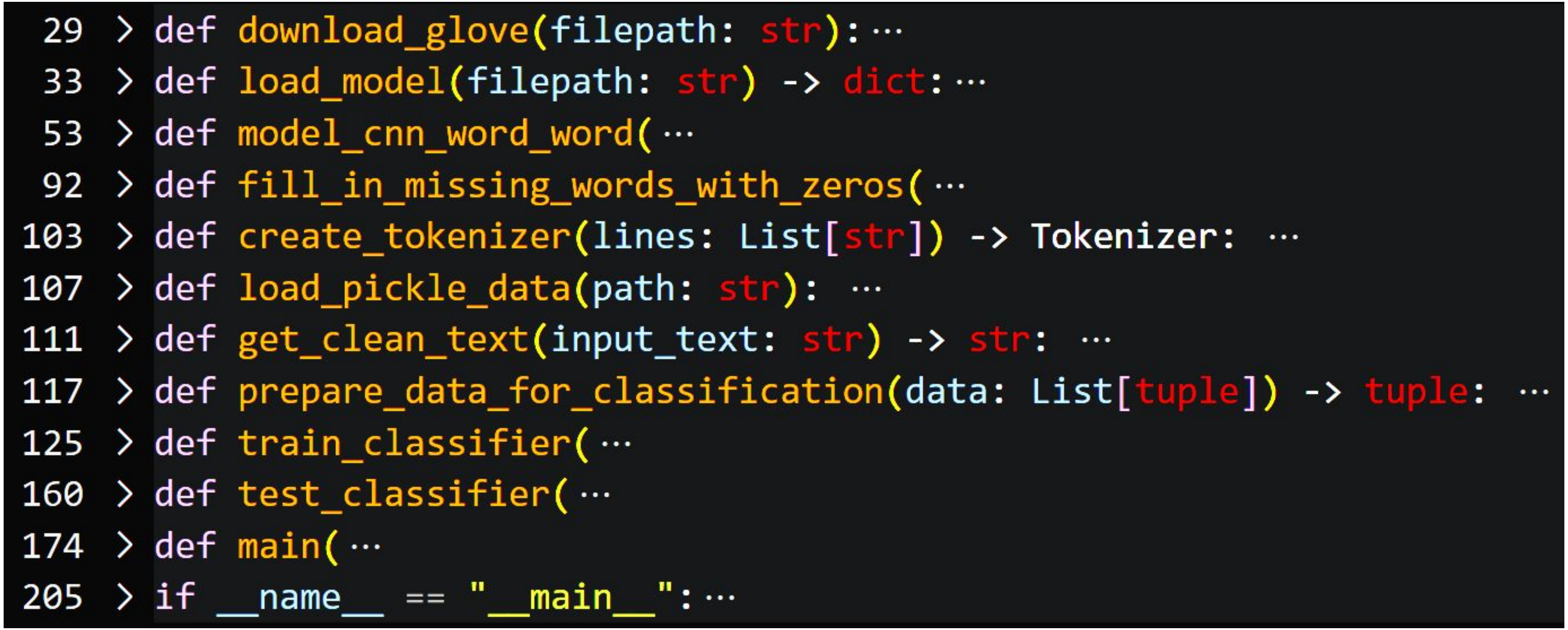
Multi-channel CNNs extend the capabilities of standard CNNs by processing data with multiple input channels simultaneously. Each channel represents different types of information, and the layers operate across all channels simultaneously, allowing the network to learn features that integrate information from each channel.



| Layer | Output Shape | Number of Parameters | Number of Input Channels | Number of Output Channels |
|---|---|---|---|---|
| Input | (12998) | 0 | 2 | 2 |
| Embedding | (12998,300) | 9331200 | 2 | 2 |
| Concatenate | (12998,600) | 0 | 2 | 1 |
| 1D-Convolution | (12994,64) | 192064 | 1 | 1 |
| 1D-Max Pooling | (1,64) | 0 | 1 | 1 |
| Flatten | (64) | 0 | 1 | 1 |
| Dense | (256) | 16640 | 1 | 1 |
| Dropout | (256) | 0 | 1 | 1 |
| Dense | (5) | 1285 | 1 | 1 |



# Implementation



```
def download_glove(filepath: str):…
def load_model(filepath: str) -> dict:…
def model_cnn_word_word(…
def fill_in_missing_words_with_zeros(…
def create_tokenizer(lines: List[str]) -> Tokenizer: …
def load_pickle_data(path: str): …
def get_clean_text(input_text: str) -> str: …
def prepare_data_for_classification(data: List[tuple]) -> tuple: …
def train_classifier(…
def test_classifier(…
def main(…
if __name__ == "__main__":…
```

Collapsed view showing code structure for implementing the suggested approach.

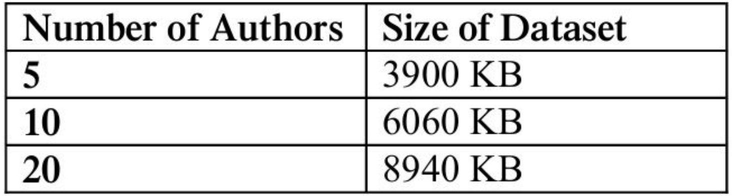
# Implementation

- **download_glove(...)** checks for the presence of the GloVe model in the local directory and automatically downloads the model only if not found, since it is a large file and can significantly slow down the execution.
- **load_model(...)** loads the word embeddings from the GloVe model.
- **model_cnn_word_word(...)** defines a CNN model with two parallel embedding layers (trainable and non-trainable), and pass the word embeddings, word indices, dimensions of word embeddings, etc. as parameters.
- **fill_in_missing_words_with_zeros(...)** takes in a dictionary of word embeddings and a dictionary of word indices, and returns a matrix of word embeddings where any missing words are filled in with zeros.
- **create_tokenizer(...)** creates a tokenizer object and fit it on the given lines of text, and returns a tokenizer object that has been fit on the given text.

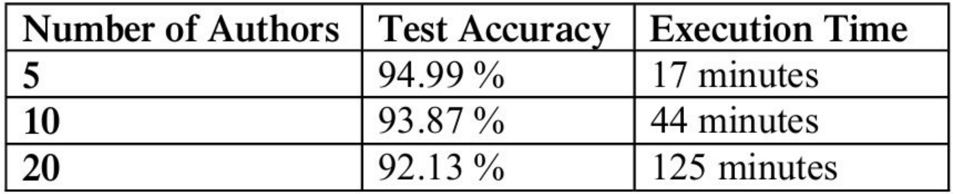
# Implementation

- **load_pickle_data(...)** loads data from a pickle file, and returns data loaded from the pickle file.
- **get_clean_text(...)** cleans the provided text by removing special characters and converting the text to a sequence of words, and returns cleaned text string.
- **prepare_data_for_classification(...)** extracts the features for input text and author label from each row, and returns list of tuples containing the cleaned text and the author id.
- **train_classifier(...)** trains a classifier using the provided training data, and returns trained classifier and StandardScaler used to scale the data.
- **test_classifier(...)** tests the classifier using the test data.

# Results and Analysis



| Number of Authors | Size of Dataset |
|---|---|
| 5 | 3900 KB |
| 10 | 6060 KB |
| 20 | 8940 KB |



| Number of Authors | Test Accuracy | Execution Time |
|---|---|---|
| 5 | 94.99 % | 17 minutes |
| 10 | 93.87 % | 44 minutes |
| 20 | 92.13 % | 125 minutes |

Test accuracy values drop faster as dataset size increases but remain **above 90%**, as we increase the number of epochs accordingly.

Using a multi-channel CNN yields **higher accuracy values** but it also **consumes more time** for similar dataset sizes. Thus, there appears to be an increasing trade-off between effectiveness and efficiency as we increase the dataset size.

# Results and Analysis

| Test Accuracies | | Linguistic Feature Sets (Optimal Cases Highlighted) | | | | |
|---|---|---|---|---|---|---|
| | | Count | TF-IDF | Writeprint | Basic-9 | Embedding |
| ML Algorithms | Logistic Regression | < 70.2% | - | - | - | - |
| | PCA & RF | - | < 74.8 % | - | - | - |
| | SVM | - | < 75.2 % | - | - | - |
| | KNN | - | < 79.5 % | - | - | - |
| | AMNP & Naive Bayes | < 80.9 % | - | - | - | - |
| | Naive Bayes | - | < 81.0 % | - | - | - |
| | CNN | < 84.2 % | - | - | - | - |
| | Multi-channel CNN | - | - | - | - | > 90 % |

# Scope for Future Work

- **Vernacular/Multilingual Environments -** Identification of language agnostic features for improved cross-language attribution.
- **Demographic Identification -** Attribution of demographic specifics of the author (age, gender, country, education, profession, personality type, political views, etc.) when list of all authors is not available.
- **Dynamic and Evolving Models -** Ability of models to adapt to changes in writing styles and patterns over time for long-term authors.
- **Real-time Authorship Attribution -** Authorship attribution during text generation to enhance real-time monitoring of online content.
- **Ethical Considerations and Bias Mitigation -** Addressing ethical issues to ensure responsible and unbiased use of attribution.

# Conclusion

Authorship attribution on news and blog datasets using multi-channel convolutional neural networks (CNNs) with global vector word embeddings **outperforms traditional machine learning methods** which use other linguistic feature sets for authorship attribution.

There is **room for improvement** in the efficiency of the approach. Also, there is scope for future research in the applicability of this approach in vernacular or multilingual environments, dynamic or evolving writing styles, real-time author identification, sociolinguistic or demography-based analysis, and bias mitigation in attribution.

# References

- Sreenivas Mekala, Vishnu Vardan Bulusu, and Raghunadha Reddy T., **“A Survey On Authorship Attribution Approaches”** in *International Journal of Computational Engineering Research (IJCER)*, vol. 8, issue 9, pp. 2250 – 3005, 2018.
- Moshe Koppel, Jonathan Schler, and Shlomo Argamon, **“Authorship Attribution: What’s Easy and What’s Hard”** in *Journal of Law and Policy*, vol. 63, pp. 23 – 68, 2013.
- Abdulaziz Altamimi, Saud Alotaibi, and Abdulrahman Alruban, **“Surveying the Development of Authorship Identification of Text Messages”** in *International Journal of Intelligent Computing Research (IJICR)*, vol. 10, issue 1, pp. 109 – 121, 2019.
- Sicong Shao, **“Machine Learning-Based Author Identification for Social Media Forensics”**, *University of Arizona*, pp. 32 – 45, 2021.
- S. M. Nirkhi, and R. V. Dharaskar, **“Comparative Study of Authorship Identification Techniques for Cyber Forensics Analysis”** in *International Journal of Advanced Computer Science and Applications*, pp. 32 – 35, 2013.

# References

- Amelec Viloria, Omar Bonerge Pineda Lezamab, and Eduardo Chang, **“Classification of Authors for an Automatic Recommendation Process for Criminal Responsibility”**, *The 7th International Symposium on Emerging Inter-networks, Communication and Mobility (EICM)*, Leuven, Belgium, 2020.
- Łukasz Gągała, **“Authorship Attribution with Neural Networks and Multiple Features”** in *Notebook for PAN at CLEF 2018*, 2018.
- Julian Hitschler, Esther van den Berg, Ines Rehbein, **“Authorship Attribution with Convolutional Neural Networks and POS-Eliding”**, in *Proceedings of the Workshop on Stylistic Variation, Association for Computational Linguistics*, pp. 53 – 58, 2017.
- Dylan Rhodes, **“Author Attribution with CNN: Overview of the Author Identification Task”** at *PAN 2014*, analysis, pp. 13 – 31, 2014.
- Javier Calle-Martín, and Antonio Miranda-García, **“Stylometry and Authorship Attribution”** in *Introduction to the Special Issue, English Studies*, pp. 251 – 258, 2012.

# Thank You

THEQUICKBROWNFOXJUMPS
OVERTHEL
BROWNF
VERTHEL
AZYDOGT
ROWNFO
XJUM
OV
AZYDOGT
XJUMPSO
VERTH
HEQUICKB
ROWNFOX
VERTHELA
ZYDOGTHEQUICKBROWNFOX