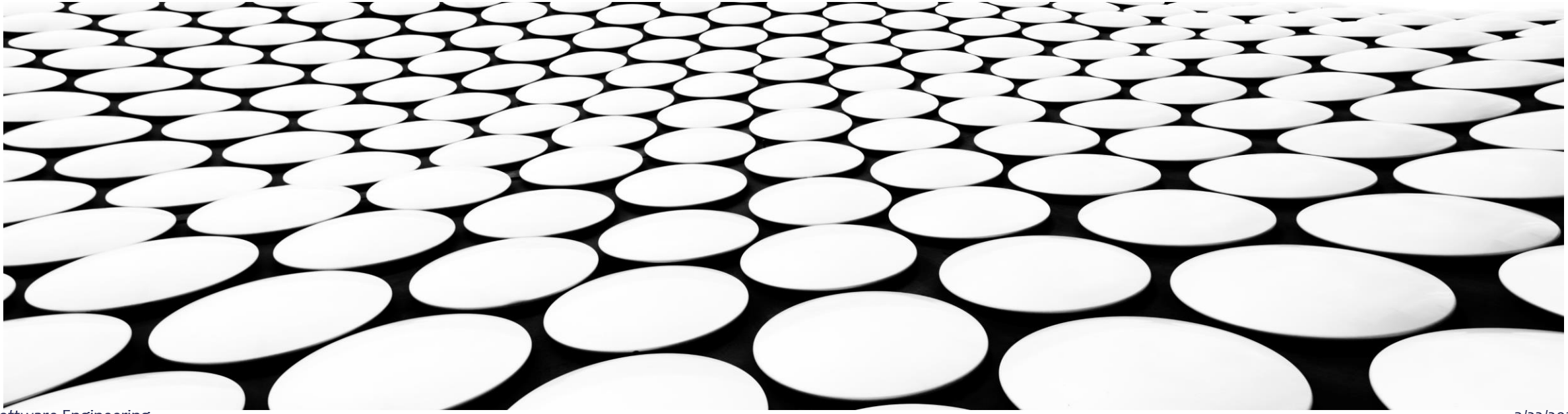
SEMANTIC ANALYSIS OF TEXTUAL DATA

Presented by team Tech_Tweakers,
Neethu Ninan-1502866
Senthil Arumugam Ramasamy-1565724

Guided by,
Prof. Damir Dobric



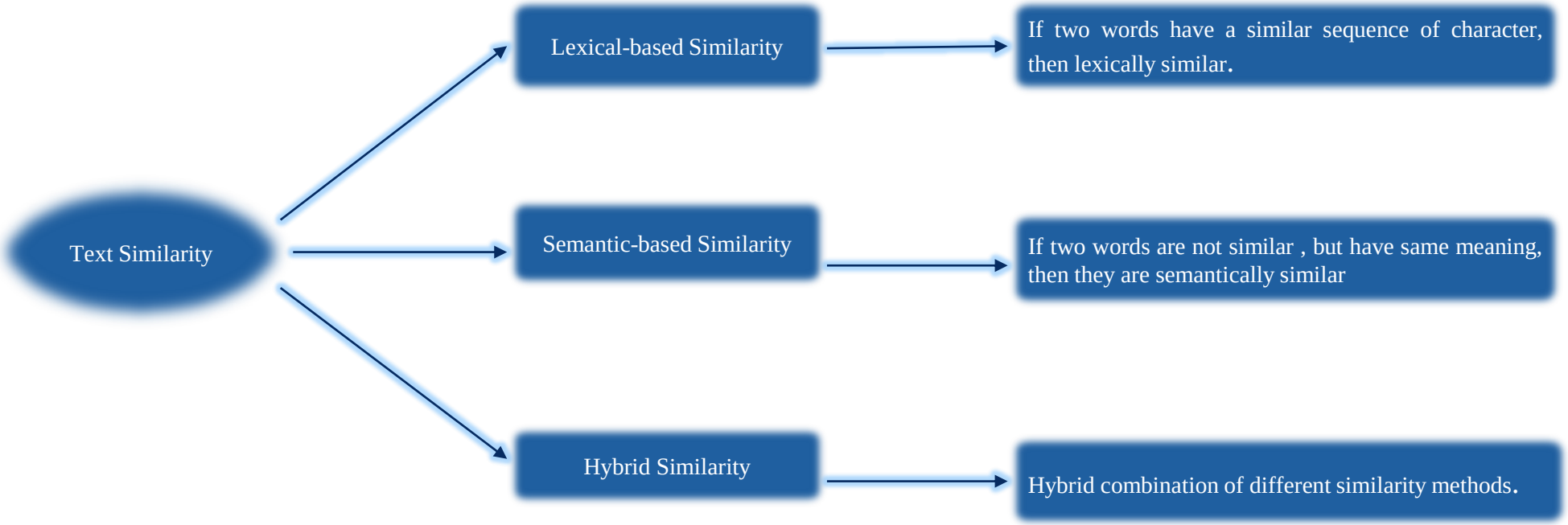
JOURNEY THROUGH THE PRESENTATION

- Objective
- Types of Textual Similarity
- What is Semantic Similarity?
- Why Open AI GPT?
- Building the Semantic Analysis Framework?
- Document Comparison Analysis
- Embedding and Similarity Calculation
- Visualization
- Test Execution Method
- Results
- Overcoming Limitations & Future Scope
- Conclusion
- References

OBJECTIVE

* Develop a scalable framework for Semantic Analysis of Textual Data focusing on words/phrases and/or documents comparisons using OpenAI embeddings and Cosine Similarity Algorithms.
* The tool offers flexible preprocessing, ensuring contextually relevant similarity assessments.
* It demonstrates effectiveness in capturing semantic nuances with practical applications like resume filtering, admission categorization, and content classification.
* Visual tools enhance understanding of similarity metrics, promoting further research in natural language processing.

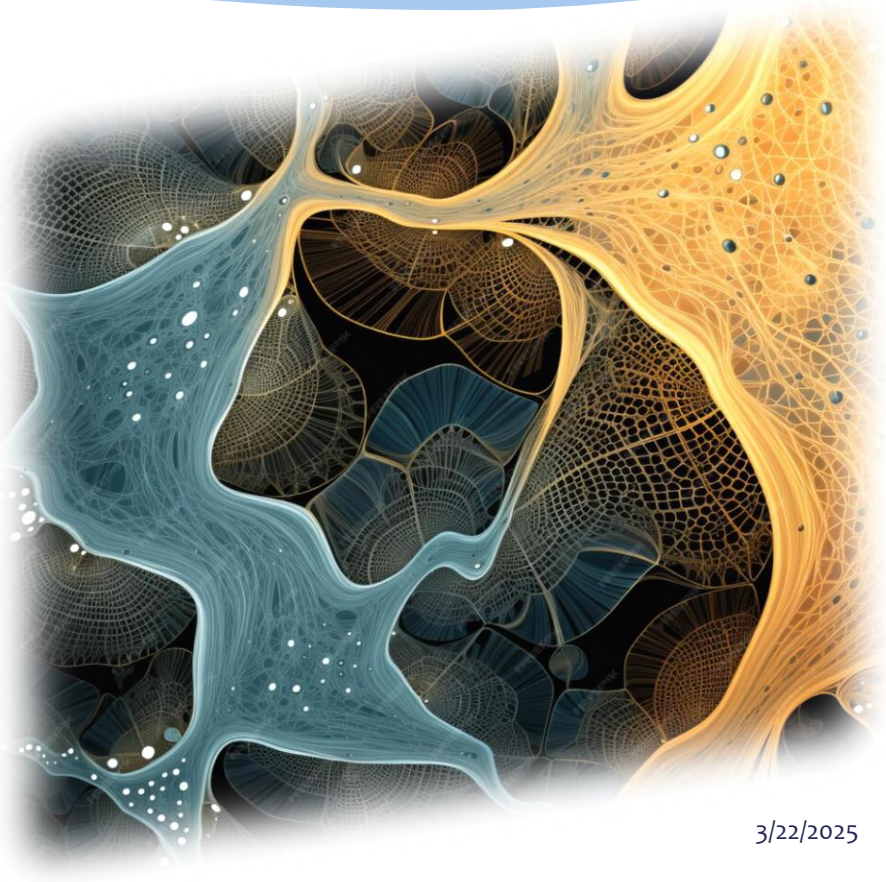
TYPES OF TEXTUAL SIMILARITY



WHAT IS SEMANTIC SIMILARITY?

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

* Measures similarity based on semantic content, not just word matching.
* Uses distance between objects to assess similarity.
* Identifies direct and indirect relationships between documents or phrases.
* A key challenge in Natural Language Processing (NLP).Information retrieval, clustering and recommendation systems are crucial NLP jobs
* Evaluates similarity using meaning rather than just words
* Assess both direct and indirect connections between texts



WHY CHOOSE OPEN AI'S GPT?

Classical Methods:
LSA(Latent Semantic Analysis) and ESA(Explicit Semantic Analysis)

Distributed Methods:
Word2Vec , GloVe.

Transformer-based Methods-
BERT/S-BERT, RoBERTa, OpenAI GPT Model, DeCLUTR

Open AI-GPT Model

- Creates dynamic embeddings that are context sensitive
- Efficiently conveys polysemous meanings.
- Less fine- tuning due to robust pre-training.
- Enhances generalization across different fields
- Dedicated embedding model converts text into high dimensional vector space.
- To use Open AI' s GPT model, subscription is needed to use the key and is based on tokens.
- Similarity calculation on the generated embeddings to yield the contextual similarity- Cosine Similarity is preferred.

BUILDING THE SEMANTIC ANALYSIS FRAMEWORK

Initial Research & Dataset Preparation:

- **Classification by Domains:**
  - Words classified into domains for meaningful analysis (e.g., Python→ Technology) Ex: Python~Programming
  - Comparison intended to check if OpenAI Embedding captures contextual relevance.

- **Dataset Preparation:**
  - 50 to 60 words from various domains prepared for comparison.
  - Editable CSV format file for dynamic data modification by developers.

Words Comparison Analysis

- **Objective:**
  - Compare words from similar and different domains to evaluate contextual relevance.
  - Ensure comparison technique effectively identifies relatedness between words within domains.

DOCUMENT COMPARISON ANALYSIS

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Objective:

- Comparing documents to establish contextual relevance between them.
- Example Use Case:
  - Source Document: JobRequirement.txt
  - Target Documents: Job Profile A, Job Profile B

Dynamic Document Comparison:

- Allows users to add documents via File Manager or custom UI integration.
- Application supports dynamic comparison for usability and research.

Threshold Management

Purpose:

- Define a threshold while visualization for meaningful similarity measurement.
- Allow application admin to manage thresholds for different use cases.

EMBEDDINGS AND SIMILARITY CALCULATION

Generating Embeddings:

- Interface: CalculateEmbeddingAsync
- Purpose: Processes text inputs & generates embeddings using OpenAI Embeddings.
- Method: GenerateEmbeddingsAsync() returns embeddings as collections of size 3052.

Embedding Analysis:

- Custom Method: PrintScalarValues(float[] embedding) for debugging and visualization.
- Helps in understanding similarity score vs. scalar values.

Similarity Calculation:

- Calculate Similarity(float[] embedding1, float[] embedding2)
- Uses Cosine Similarity Algorithm:
  - 1: Identical embeddings, 0: No similarity, -1: Complete dissimilarity.
- Formula: Cosine Similarity = Dot Product / (Magnitude1 * Magnitude2).

VISUALIZATION

Purpose:

* Understand Semantic Analysis through visual representation.
* Correlate similarity scores to real-time use cases.

Python Integration:

* Python used as an external tool to generate graphical charts from CSV outputs.
* Current limitation: Manual placement of CSV files in the Python app's root directory.
* Future scope: Automating CSV file placement.

Types of Plots:

* Comparison Plot:
  * X-Axis: Document/Phrase Comparisons (User-designed datasets).
  * Y-Axis: Similarity Scores.

Similarity vs. Scalar Plot:

- X-Axis: Scalar Values (Ranges between 0-3052).
- Y-Axis: Similarity Scores.

Visualize how contextual relevance is generated.

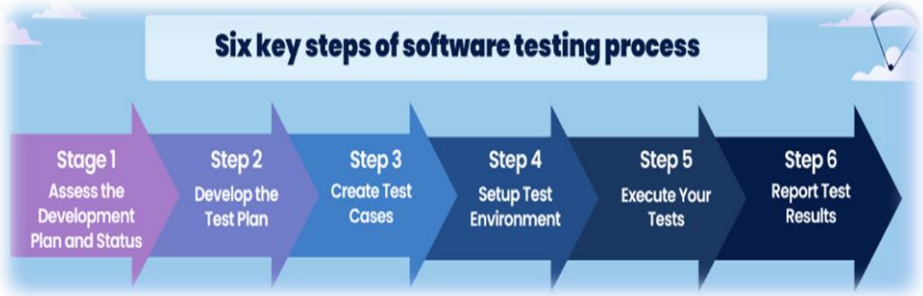
Visualization Purpose:

- Helps developers and users comprehend how embedding's map to similarity scores.
- Provides insights for improving the similarity analysis approach.

TEST EXECUTION METHODS

Purpose:

- Ensure robustness of application by validating various functionalities through test cases.
- Maintain high code coverage by testing positive, negative, and edge case scenarios.
- Test Framework:
- Implemented using Microsoft.VisualStudio.TestTools.UnitTesting.
- Tests run via Visual Studio Test Explorer.
- Test classes created for each service based on business functionalities.



Test Case Categories:

- Basic Test Framework Validation
- Exception Handling in Document Comparison
- Validity of Similarity Score Calculation
- Service Provider Configuration Validation
- Source & Target File Retrieval
- Scalar Value Printing Validation
- Accuracy of Similarity Score Calculation
- Handling of Different Length
- Handling of Empty
- Handling of Invalid File Paths
- Phrase Processing & Result Saving
- CSV Helper utility Test

RESULTS

* Results are Published using Github Pages.
* Phrase Comparison Output CSV Path - bin\Release\net9.0\data\output_datasetphrases.csv
* Document Comparison Output CSV Path - bin\Release\net9.0\data\output_dataset.csv

Phrase Comparison Results:
https://senthilmasters2024.github.io/Tech_Tweakers/PhrasesSimilarityClassficationByDomainsPlots.html

Document Comparison Results:
https://senthilmasters2024.github.io/Tech_Tweakers/SemanticSimilarityLatestPlot.html

Scalar Values of Embedding vs. Similarity Score:
https://senthilmasters2024.github.io/Tech_Tweakers/ScalarValuesVsSimilarityScorePlotForOneComparsion.html

OVERCOMING *LIMITATIONS & FUTURE SCOPE*

Limitations:

- **Dependency on External Libraries:** Changes in Plotly.NET and Microsoft.VisualStudio.TestTools.UnitTesting may affect functionality.
- **Text Preprocessing Scope:** Limited handling of complex text variations like HTML tags and nested URLs.
- **Performance:** Inefficiencies in processing large datasets.
- **Customization:** Preprocessing rules are hardcoded, limiting user flexibility.

Proposed Solutions:

- Regular updates, adapter patterns, and compatibility tests for external libraries.
- Advanced preprocessing techniques and modular functions for customization.
- Implementation of parallel processing and memory-efficient algorithms.
- Providing user-defined configuration files and settings for more adaptability.

CONCLUSION

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

* Bridging the gap between theoretical concepts and real-world applications.
* Visualizations demonstrate contextual alignment of texts and similarity scores.
* Application adaptability for job profiles, student profiles, and document comparison.
* Future Enhancements:
  - Automating CSV file integration.
  - Expanding dataset support.
  - Improving user interface for better usability.
* Contribution to NLP applications and potential for future research.

REFERENCES

[1] A. a. H. T. Aboelghit, “Textual Similarity Measurement Approaches: A Survey (1),” *The Egyptian Journal of Language Engineering*, vol. 7, no. 2, pp. 41-62, 2020.

[2] D.P.P.A.G.D.P. Goutam Man Majumder, “Semantic Textual Similarity Methods, Tools, and Applications: A Survey, Computacion Sistemas, 2016.

[3] P. Wiemer-Hastings, “Latent Semantic Analysis,” Citeseer, 2004.

[4] E. G. a. S. Markovitch, “Computing Semantic Relatedness usin Wikipedia-based Explicit Semantic Analysis,” in *International Conference on Intelligent Text Processing and Computational Linguistics*, 2013.

[5] K. C. G. C. J. D. Tomas Mikolov, “Efficient Estimation of Word Representations in Vector Space,” in

International Conference on Learning Representations, 2013.

[6] R. S. C. M. Jeffrey Pennington, “GloVe: Global Vectors for Word Representation,” in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 2014.

REFERENCES(Continued…)

[7] I. G. Nils Reimers, Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, Hong Kong, China: Association for Computational Linguistics, 2019.

[8] M.-W. C. K. L. K. T. Jacob Devlin, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota, Association for Computational Linguistics, 2019, pp. 4171--4186.

[9] https://openai.com/index/new-embedding-models-and-api-updates/

[10] https://learn.microsoft.com/en-us/azure/cosmos-db/gen-ai/vector-embeddings

[11] https://openai.com/index/introducing-text-and-code-embeddings/

[12] N. K. K. ,. K. S. Dipendra Prasad Yadav, "Distance Metrics for Machine Learning and it's Relation with Other Distances.," Mikailalsys Journal of Mathematics and Statistics, vol. 1, pp. 15-23, 2023.

[13] O. N. B. W. G. B. John Giorgi, "DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations," Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the

11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), vol. 1, p. 879–895, 2021

TEAM,
TECH_TWEAKERS