ML Challenge – Movie Similarity Model

Research & Interactive Demo

Albert Sallés

February 17, 2025

Overview

1. Semantic Similarity
2. Project Code
3. Deployment & Infrastructure

Semantic Similarity

Semantic similarity techniques can be divided into four categories:

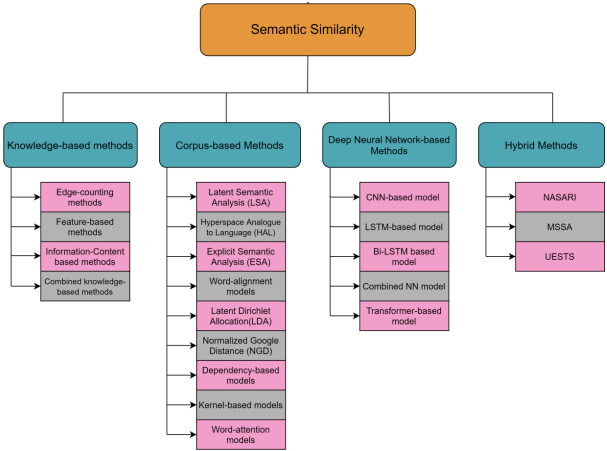

Figure: Semantic Similarity Methods [Chandrasekaran and Mago, 2021]

Semantic Similarity – Knowledge-based methods

These methods calculate the similarity based on a structured lexical resource like WordNet or ConceptNet.

- **Edge counting**: Counts the number of edges in the shortest path between two concepts.
- **Feature-based**: Using a set of attributes about two concepts, measures similarity based on the shared and distinct attributes.

Movie Recommendation

These techniques only offer semantic similarity between two words, not texts.

Semantic Similarity – Corpus-based methods

They use statistical analysis of a large text corpus. The actual meaning of the word is not taken into consideration.

- **Bag of Words**: each word is a feature in a vector indicating the frequency of the word occurrences.
- **Term Frequency-Inverse Document Frequency**: enhances BoW by considering the rarity of each word across the text.
- **Word embeddings**: Vector representations pre-computed using different techniques like neural networks or word co-occurences matrix from a large corpus: Word2Vec, GloVe, FastText.

Movie Recommendation

These are the most efficient methods since they operate on pre-computed or simple operations.

Semantic Similarity – DNN-based methods

Transformer-based methods gather context information and weigh words importance based on attention mechanism. [Vaswani, 2017]

- **Tranformer Embeddings**: They outperform most of the traditional methods. Their performance is increased by using larger corpus. However, they lack interpretability.

Movie Recommendation

While these methods offer a more context and word meaning aware system, the computational cost of calculating the embeddings is higher.

Embeddings – Similarity Distance Measure

Embeddings are vector representations of words or texts. To compute how close two embeddings are, there exist many metrics:

- **Cosine similarity**: Measures the angle between two vectors.
- **Euclidean distance**: Absolute distance between two points in the space.
- **Dot product**: Measures vector alignment.

Popular choice

Cosine similarity has become the most popular choice because of its scale-invariance property. It only takes into account the direction of the vector. Thus, direction in the vector space represents similarity.

Future Improvements – Similarity Distance Measure

Using Word2Vec is a middle ground between simple corpus and DNN-based approaches. However, the meaning between words in different contexts are not taken into account:

- Words like *lake* mean the same in *mountain lake* and *data lake*.
- Sentences that contain the same words in different orders have the same embeddings.

This simple movie recommendation system does not require such complex structures. A future improvement would be to add a simple transformer embedding.

Project Structure

The architecture is based on the interface-service-repository pattern to ensure scalability, maintainability and independence of layers:

1. **Interface Layers (API)**: Acts as the entry points for client requests and handles external communication.
2. **Service Layer**: Contains the business logic.
3. **Repository Layer**: Handles data access, ensuring separation between business logic and data persistence.

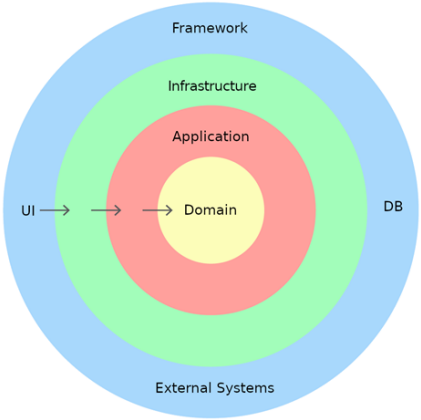

Figure: Repository Pattern

Deployment

The application has been Dockerized so there are many options where it can be deployed:

- Server: run a custom EC2 instance with Docker Compose.
- Serverless: AWS Lambda/Fargate for fully-managed container and scaling. The Docker image can be pushed to ECR and the deployment is automatized.

There are key differences when choosing a serverless option:

AWS Lambda

- For lightweight, short tasks
- Stateless (no persistance)
- Auto-scaling
- Pay per invocation/execution time.

AWS Fargate

- Runs Docker containers.
- Handles long-running tasks.
- Auto-scaling (higher startup time)
- Costs more for lightweight tasks.

References

Chandrasekaran, D. and Mago, V. (2021).
Evolution of semantic similarity—a survey.
ACM Computing Surveys (CSUR), 54(2):1–37.

Vaswani, A. (2017).
Attention is all you need.
Advances in Neural Information Processing Systems.