

# IRSE Project

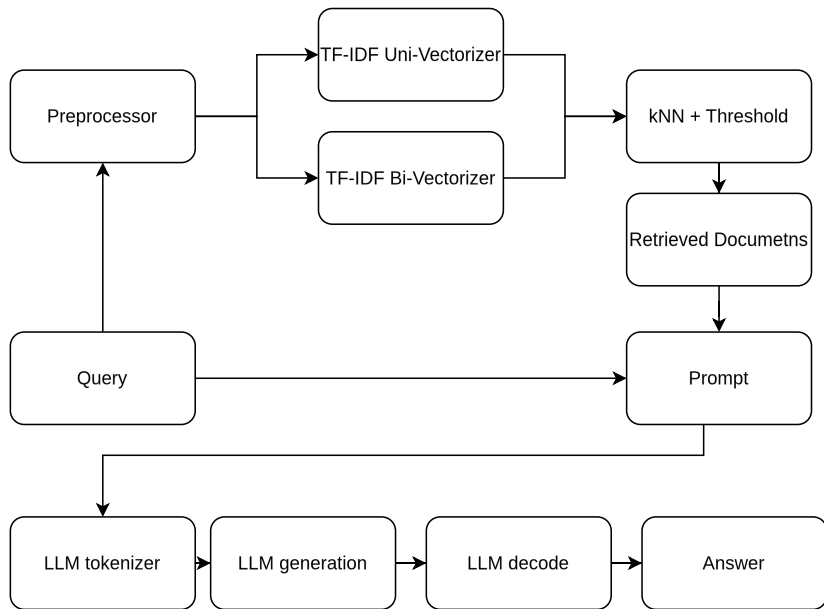
Jan Cichomski (r1026448)

May 14, 2025



- why have i chose F1 as a metric?

# 1. Architecture



## 2. Term Vocabulary

## 2.1 Term Vocabulary - Document Preprocessing

- To lower case
- Remove punctuation
- Tokenize
- Remove english stop words (added custom stop words)
- Lemmatize

## 2.1 Temr Vocabulary - Document Preprocessing

```
def preprocess_text(doc):  
    doc = doc.translate(str.maketrans("", "",  
        string.punctuation)).lower()  
    words = word_tokenize(doc)  
    words = [  
        lemmatizer.lemmatize(word)  
        for word in words  
        if word not in stop_words and word.isalpha()  
    ]  
    return " ".join(words)
```

## 2.1 Term Vocabulary - Custom stop words

```
stop_words.update(  
    [  
        "add",  
        "added",  
        "adding",  
        "addition",  
        "also",  
        "almost",  
        "another",  
        "easily",  
        "easy",  
    ]  
)
```



## 2.2 Term Vocabulary - Hyperparameters

Two types of terms:

- 1-grams
  - min\_df=20
  - max\_df=0.5
- 2-grams
  - 10,000 terms
  - min\_df=50
  - max\_df=0.4

## 2.3 Term Vocabulary - Handling multi-word terms

- 2-grams with aggressive filtering

# 3 Document Embedding

## 3.1 Document Embedding - Chosen Fields

Fields used for embedding:

- name
- description
- ingredients
- steps

Evaluated combinations:

- name, description, ingredients, steps: macro F1: 0.126
- description, ingredients, steps: macro F1: 0.095
- description, ingredients: macro F1: 0.034
- description, steps: macro F1: 0.088
- description: macro F1: 0.043

## 3.2 Document Embedding - Query Preprocessing

The same approach as for embedding documents

```
def retrieve_documents(query_text, recipes,  
                      recipe_ids, k, threshold):  
    query = preprocess_text(query_text)  
    ...
```

## 3.3 Document Embedding - Edge Cases

- When query has no terms from vocabulary
  - TF-IDF produces zero vector for the query
  - Cosine similarity returns 0 for all documents
- Consequently:
  - Without similarity threshold: All documents returned (no filtering)
  - With any similarity threshold: No documents returned (empty result)

## 4 Retrieval

## 4.1 Retrieval - Similarity Measure

- Cosine similarity (F1: 0.126) - picked finally
- Euclidean distance (F1: 0.064)



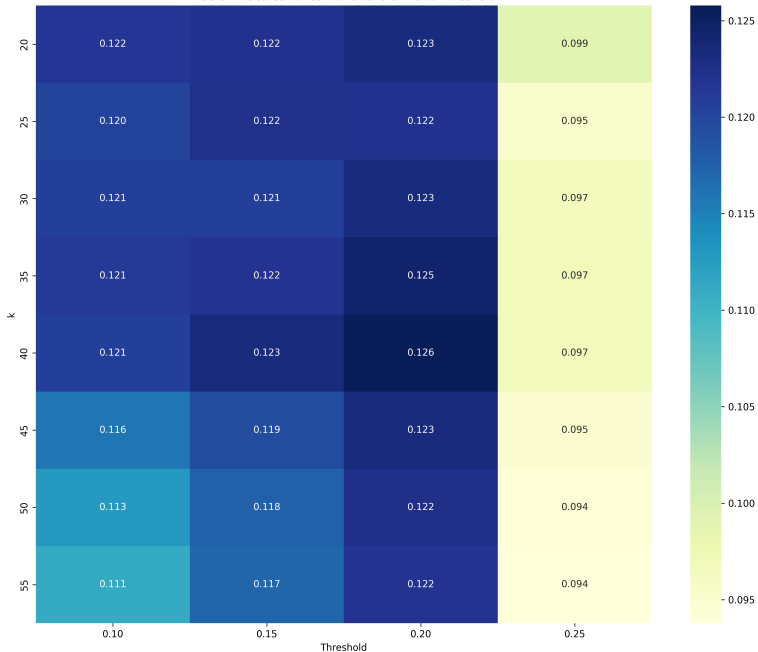
## 4.2 Retrieval - Hyperparameters

- Max number of returned documents: 40
- Minimum threshold for cosine similarity: 0.2

Grid search over param space and maximize macro F1 score.

```
def create_parameter_heatmap(queries, recipes, recipe_ids):  
    thresholds = np.arange(0.1, 0.60, 0.05)  
    k_values = np.arange(20, 60, 5)
```

Macro F1 Scores for Combinations of k and Threshold



## 4.3 Retrieval - Evaluation Metrics

- Macro Precision: 0.130
- Macro Recall: 0.201
- Macro F1: 0.126
- Micro Precision: 0.128
- Micro Recall: 0.191
- Micro F1: 0.153

## 4.4 Retrieval - MAP

Mean Average Precision (MAP): 0.086

$$AP = \frac{1}{RD} \sum_{k=1}^n P(k) \cdot r(k), \quad (1)$$

Where  $RD$  is the number of relevant documents for the query,  $n$  is the total number of documents,  $P(k)$  is the precision at  $k$ , and  $r(k)$  is the relevance of the  $k^{th}$  retrieved document (0 if not relevant, and 1 if relevant)

$$MAP = \frac{1}{Q} \sum_{i=1}^Q AP_i \quad (2)$$

Where  $Q$  is the number of queries and  $AP_i$  is the average precision for the  $i^{th}$  query.

## 4.4 Retrieval - MAP Code

```
def calculate_average_precision(relevant_doc_ids,
                                retrieved_doc_ids):
    hit_count = 0
    sum_precisions = 0.0
    for i, doc_id in enumerate(retrieved_doc_ids):
        if doc_id in relevant_doc_ids:
            hit_count += 1
            precision_at_i = hit_count / (i + 1)
            sum_precisions += precision_at_i
        # else: sum_precisions += 0.0
    if len(relevant_doc_ids) == 0:
        return 0.0
    return sum_precisions / len(relevant_doc_ids)
```

## 5. Qualitative analysis - Information Retrieval

TODO TODO TODO

## 5.1 Qualitative analysis - Information Retrieval

Problem: Even though there is no relevant information in the document, the system returns some documents

Prompt: "Where can I follow cooking classes"

Output: cinnamon roll glaze taste facts class (0.2410), grandma jayne shrimp dip (0.2369)

## 5.2 Qualitative analysis - Information Retrieval

Problem: Ignores context of entities in query

Prompt: "How does Gordon Ramsay make his beef Wellington?"

Output: Non of the results were about Ramsay making beef wellington

ID	Score	Words Contained	Words Not Contained
94359	0.2972	ramsay, gordon	beef, wellington
94358	0.2502	ramsay, gordon	beef, wellington
111233	0.2448	beef, wellington	ramsay, gordon
163842	0.2439	beef, wellington	ramsay, gordon
94347	0.2207	ramsay, gordon	beef, wellington
100473	0.2146	beef, wellington	ramsay, gordon
126542	0.2086	beef, wellington	ramsay, gordon
94354	0.2069	ramsay, gordon, beef	wellington
170428	0.2032	ramsay, gordon	beef, wellington
94353	0.2029	ramsay, gordon	beef, wellington



## 5.3 Qualitative analysis - Information Retrieval

Problem: Can't handle external rare words, like "Paraguay"

Prompt: "Do you know any soups from Paraguay?"

Output: Did not returned any recipe with word "paraguay"

## 5.4 Qualitative analysis - Information Retrieval

Problem: TF-IDF doesn't handle typos

Prompt: "How do you make **piza**"

Output: Returned single recipe what contains a lot of words "make", but no "pizza"

## 5.5 Qualitative analysis - Information Retrieval

Problem: Can't capture negation

Prompt: "I do not want to eat pizza, what can I eat instead?"

Output: 36/40 results were about pizza

## 6. Prompt

Model: mistralai/Mistral-7B-Instruct-v0.2

## 6.1 Prompt - LLM Instructions - Good

- General context and LLM's goal
- Instructions per type of question
- Response format
- Limitations

See handout for full prompt TODO: add full prompt in handout

## 6.1 Prompt - LLM Instructions - Bad

Only general context without specific instructions how to answer the question

You are a helpful recipe assistant with access to a database of recipes. The system has already retrieved the most relevant recipes to the user's query using TF-IDF similarity. Your goal is to provide helpful, accurate responses about recipes, cooking techniques, ingredient substitutions, and culinary advice based on the retrieved recipes.

The following recipes have been retrieved as most relevant to the user's query:

{retrieved\_recipes}

## User Query

{user\_query}

## 6.2 Prompt - Fields used

- name
- description
- ingredients
- steps
- relevance score

## 6.2 Prompt - Fields used

```
results = retrieve_documents(query, recipes, recipe_ids, k=k,  
retrieved_recipes = ""  
  
for idx, (recipe, recipe_id, score) in enumerate(results):  
    info=df[df["official_id"] == recipe_id].iloc[0]  
    retrieved_recipes+=f"Document {idx}, Score: {score:.4f}\n"  
    retrieved_recipes+=f"Name: {info['name']}\n"  
    retrieved_recipes+=f"Description:{info['description']}\n"  
    retrieved_recipes+=f"Ingredients:{info['ingredients']}\n"  
    retrieved_recipes+=f"Steps: {info['steps']}\n\n"  
  
prompt = prompt_template.format(  
    retrieved_recipes=retrieved_recipes, user_query=query  
)
```



## 7. Qualitative analysis - LLM

## 7.1 Qualitative analysis - LLM & RAG

- More or less yes
- But does not stick to rules
- Response are often too general

## 7.2 Qualitative analysis - LLM Reasoning 1

Q: "I'm looking for a spicy pasta sauce, but I forgot the name. I think it was related to Arabs?" TODO

## 7.2 Qualitative analysis - LLM Reasoning 2

TODO

## 7.2 Qualitative analysis - LLM Reasoning 3

TODO

## 7.2 Qualitative analysis - LLM Reasoning 4

TODO

## 7.2 Qualitative analysis - LLM Reasoning 5

TODO

## 7.3 Qualitative analysis - Hallucination

- Standard prompt, but with no documents provided  
"How does Gordon Ramsay make his beef Wellington?"
- Yet LLM answered with recipe
- Which suggest that it did not followed rules

The following recipes have been retrieved as most relevant to the user's query:

## Instructions

{rest of the prompt}

[/INST] Based on the retrieved recipes, Gordon Ramsay's Beef Wellington is typically made with a large piece of beef fillet, duplicated with a sheet of pate, covered in mushrooms and pastry, and baked in an oven.

...



## 7.4 Qualitative analysis - Score vs No Score - TODO

## 8. Neural document embeddings

## 8.1 Neural document embeddings - out of vocabulary - query

Used model's tokenizer to tokenize the query

- Original text: 'kashubian'
- [CLS] ', 'ka', '##shu', '##bian', '[SEP]'

The ## prefix represents subword tokenization, allowing the model to handle words not in its vocabulary.

## 8.1 Neural document embeddings - out of vocabulary - document

Used model's tokenizer to tokenize the query

- Original text: 'their settlement area is referred to as kashubia they speak the kashubian language which is classified either...'
- '[CLS]', 'their', 'settlement', 'area', 'is', 'referred', 'to', 'as', 'ka', '##shu', '##bia', 'they', 'speak', 'the', 'ka', '##shu', '##bian', 'language', 'which', 'is', 'classified', 'either',

Neural embeddings returned valid results even through the word "kashubian" was not in the vocabulary.

## 8.2 Neural document embeddings - Metrics - Recipes

- Macro Precision: 0.307
- Macro Recall: 0.134
- Macro F1: 0.157
- Micro Precision: 0.280
- Micro Recall: 0.035
- Micro F1: 0.062
- MAP: 0.101
- Average DCG: 2.472
- Average NDCG: 0.736

## 8.2 Neural document embeddings - Metrics - Wiki

- Macro Precision: 0.310
- Macro Recall: 0.352
- Macro F1: 0.260
- Micro Precision: 0.343
- Micro Recall: 0.303
- Micro F1: 0.322
- MAP: 0.216
- Average DCG: 1.566
- Average NDCG: 0.549

## 9. Compression

# 9.1 Compression - Long Documents

## Information Retrieval

- Cover multiple topics
- Contain lots of words
- Limited document length

## LLM

- Limited context window
- Needle in a haystack



## 9.2 Compression - Solutoin

Split documents into chunks

```
from langchain_text_splitters
    import RecursiveCharacterTextSplitter
text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=500,
    chunk_overlap=100,
    length_function=len,
    is_separator_regex=False,
)
```

## 9. Security

## 9.1 Security

- Yes, LLM is susceptible TODO...