INFORMATION RETRIEVAL

Flavio Forenza
flavio.forenza@studenti.unimi.it

Department of Computer Science,
University of Milan, Italy

September 12, 2021

INTRODUCTION

Nowadays, searching the web for information appears to be one of the simplest operations to perform. The difficulty perceived by the user in formulating a query has been gradually reduced by techniques capable of guiding his writing towards a correct generation of a query. These techniques allow to improve the performance of information search systems.

GOAL

The purpose of this project is to be able to experiment with the use of one of the most famous techniques, already present at the state of the art, able to "assist" the user in formulating a correct query: **Language Modelling**. Query expansion can be done using this concept to return a corpus of relevant documents.

DATASET DESCRIPTION

The dataset used for the experiments is the famous *Recipes1M+* [1], a collection created by MIT, consisting of more than one million culinary recipes. Of all these recipes, only a subset of 51235 documents of it was used due to their informative content which best fits the purpose of this study. The information about the line distributions for each recipe indicates that the instruction field contains a higher number than the information contained in the ingredients field.

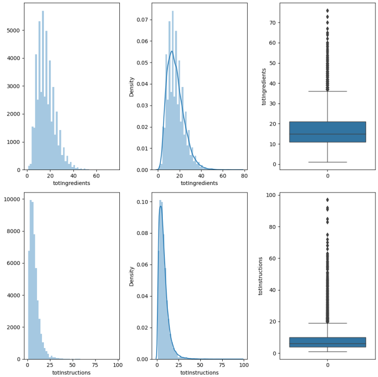

Figure. Distributions of lines per ingredients and instructions.

[1] J.Marin, A.Biswas, F.Ofli, N.Hynes, A.Salvador, Y.Aytar, I.Weber and A.Torralba, "Recipe1M+: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images", IEEE Trans. Pattern Anal. Mach. Intell., 2019

RANKING GENERATION

The first step is based on choosing a random query, which resembles the title of one of the existing recipes. Subsequently, thanks to the combination of the tf-idf method and the cosine similarity metric, it is possible to generate the first ranking of documents ordered according to relevance with the query. The threshold chosen, for the selection of the most relevant documents, will correspond to the weight assigned by the tf-idf to the target document.

tf-idf

$$TfIdf(q_t, d) = tf_{q_t,d} \log \frac{N}{df_{q_t}}$$

Cosine similarity

$$cosine(q, d) = \frac{\sum_{i=1}^{N} qd_i}{\sqrt{\sum_{i=1}^{N} q^2}\sqrt{\sum_{i=1}^{N} d_i^2}}$$

RANKING EVALUATION

The evaluation of the ranking of relevant documents was carried out considering as the entity, of each single document, its category of belonging[1] (dessert, salad, beverage, etc.). The search for each category takes place in two methods, with two different libraries:

- *Scrape Schema Recipe*
- *USDA* (United States Department of Agricolture)

Queries	Scrape Schema Recipe			USDA	Mixed (Scrape+USDA)
	Overestimate	Underestimate	Discarded		
I	0.7520	0.3734	0.5958	0.9817	0.9947
II	0.9309	0.6780	0.9054	1.0	1.0
III	0.7458	0.2746	0.5411	0.9797	0.9982
IV	0.8939	0.5320	0.8433	0.9612	0.9870
V	0.8335	0.5033	0.7544	0.9940	0.9940
Average	0.8312	0.4722	0.7280	0.9833	**0.99478**

Table: Average Precision on each query for each method.

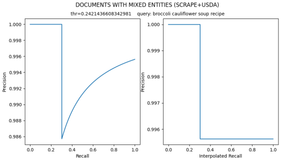

Figure. Performance in terms of Precion, Recall and Interpolated Recall.

[1] Md R.Parvez et al. "Building Language Models for Text with Named Entities". In: (2018), pp. 373–2383.

LANGUAGE MODELS

For each document, present in the ranking, having a threshold greater than or equal to the weight assigned to the target document, a language models will be constructed consisting of a sequence of *bi-grams*, with an initial *skip-grams* equal to two, with the relative count of every occurrence.

Sentence probability (Bi-gram) and MLE

$$\begin{aligned} P(q|d) &\approx P(q|M_d) \\ &\approx \prod_{i}^{n} P(w_i|w_{i-1}) \\ &\approx \frac{count(w_i, w_{i-1})}{\sum_{j=1}^{n} count(w_j, w_{i-1})} \\ &= \frac{count(w_i, w_{i-1})}{count(w_{i-1})} \end{aligned} \tag{1}$$

SMOOTHING METHODS

To avoid having a probability equal to zero, two smoothing techniques are calculated:

Laplace Smoothing

$$P(w_i|w_{i-1}) = \frac{count(w_{i-1})+1}{count(w_{i-1})+|V|}$$

Linear Interpolation Smoothing (Bi-grams)

$$P(w_i|w_{i-1}) = \lambda P(q|M_d) + (1-\lambda)P(q|M_c)$$

⋮

Linear Interpolation Smoothing (Zero-grams)[3]

$$P(w_i) = \lambda \frac{1}{|V|} + (1-\lambda)P(w_i)$$

where:

- $\sum_i \lambda_i = 1$
- M_d: LM of the single document;
- M_c: LM of the entire collection of documents;
- $|V|$: # unique words within the corpus of documents.

[3] A.Gutnik, "Log-Linear Interpolation of Language Models", 2000

CORE

Algorithm 1: Best Ranking

```
input     : q
output    : final_ranking
parameter : LMtd
for s ← 2 to 10 do
    LMdocs ← ComputeLMdocs(s, query);
    LMcoll ← ComputeLMcoll(s, query);
    rankLpl ← LplSmoot (q, LMdocs);
    perplexityLpl ← perplexityLpl (q, LMtd);
    lamb1, lamb2 ← 0;
    for i ← 0.1 to 1.0 do
        rankLinInt ← LinInt (q, LMdocs, LMcoll);
        perplexityInt ← perplexityInt (q, LMtd);
        lamb1 ← 0.1 + i;
        lamb2 ← 0.9 - i;
        if lamb2==0 then
            break;
        end
    end
end
.
.
.
Search for the minimum index of the target document;
final_ranking ← min(rankLpl[i],rankLinInt[i]);
return final_ranking
```

where:

- *q*: query
- *s*: skip-gram
- *final_ranking*: ranking containing the target document in the position closest to the first
- *LMtd*: Language model of target document
- *LMdocs*: List of the language models of each document
- *LMcoll*: language model of the entire collection of documents
- *lamb1*: λ_1
- *lamb2*: λ_2

TERM-TERM MATRIX

As explained in[2], two or more words are considered synonymous, or semantically important, if their *vectors*, represented in a multidimensional space, have a high *cosine similarity*. To find out, we need to build a matrix of terms, called *term-term matrix*, where the *co-occurrences* between the terms present in the relevant documents and in the query will be reported. We can exploit the *Language Model* of the entire corpus, of relevant documents, to derive co-occurrences.

	variation	broccoli	cauliflower	toss	substitute	...
variation	0	**1**	0	0	0	...
broccoli	0	0	**5**	0	0	...
cauliflower	0	0	0	**1**	0	...
toss	0	0	0	0	**1**	...
substitute	0	0	**1**	0	0	...
...	...	...	...	...	...	...

Table: Co-Occurrence of words in Term-Term Matrix.

[2] D.Jurafsky and J.H. "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition". In: vol. 3. 2008.

POSITIVE POINTWISE MUTUAL INFORMATIONS (PPMI)

Pointwise Mutual Information (PMI) was used to evaluate the relationship between two words. Mutual Informations (PMI).

PMI

"PMI draws on the intuition that the best way to weigh the association between two words is to ask how much more the two words co-occur in our corpus than we would have a priori expected them to appear by chance."[2]

In this case, its *"Positive"* version (PPMI) was used to be able to remove negative values.

PMI

$$PMI(x, y) = \log_2 \frac{P(x,y)}{P(x)P(y)}$$

PPMI

$$PPMI(x, y) = \max(\log_2 \frac{P(x,y)}{P(x)P(y)}, 0)$$

SINGULAR VALUE DECOMPOSITION (SVD)

The term-term matrix presents a serious problem that affects the performance of the system and the calculation of the cosine similarity: the **sparsity**. In order to reduce the sparsity, the *Singular Value Decomposition (SVD)* technique was applied.

$$A = USV^T$$

$$\mathcal{D} = U * S$$

$$\mathcal{T} = S * V^T$$

Where:

- A: a $t \times d$ Term-Term Matrix decomposed into 3 sub-matrices.
- U: a $t \times m$ matrix
- S: a $m \times n$ matrix
- V^T: a $m \times d$ matrix
- $\mathcal{D}$: a $t \times m$ matrix containing all the terms of the relevant documents
- $\mathcal{T}$: a $m \times d$ matrix containing the query terms useful for calculating the cosine similarity with the terms present in $\mathcal{D}$

QUERY EXPANSION

PERPLEXITY

SYSTEM EVALUATION WITH DIFFERENT PARSERS

CONCLUSIONS