Vertical Selection for Heterogeneous Search Engine Result Pages

Augustas Vilčinskas

Delft University of Technology
Delft, The Netherlands

ABSTRACT

Items that a user can see when he uses the general result page of a modern search engine can be categorized as verticals. Some examples of verticals are images, videos, news, shopping. Heterogeneous search engine result pages encompass result pages that contain results from different verticals. It is widely used and has been proven to improve the user experience over the result pages that only contain a list of websites. Different verticals are appropriate for each query. We study how to define, develop, and evaluate a vertical selection model, that for a query selects and presents the appropriate verticals. We give an approach for collecting a corpus of documents that represent different verticals. Later corpus documents are used as training data for query result classification. Features were extracted from the documents to train a classifier. The model that uses the Random Forest classifier and features extracted from the query itself achieved an f-score of 0.4921 on the TREC 2014 dataset. The score and the analysis of the results show that the proposed vertical selection methodology is viable. To better capture the difference between documents in different verticals, the corpus collection approach should be improved.

KEYWORDS

aggregated search, vertical selection, heterogeneous search engine result page

1 INTRODUCTION

Modern-day search engines allow users to search within one category of results. These categories are called **verticals**. When a user is searching for some information online, he can opt to only search within one vertical (images, videos, news) or use the general page which combines the results from different verticals into one page. A result page that contains elements from different verticals is called a heterogeneous search engine result page (**SERP**).

As an example take the query "dog on a bike", some will want to see videos of dogs on a bike, others images. Results to this query can be seen in figure 1. As shown, both image and video verticals are included in the search engine result page. This provides the user with easy access to the content in those verticals.

A study that was done by Bron et al. [5] further supports the use of SERPs that include different verticals. It found that including results from different verticals in one page, helps users understand what kind of content each vertical contains, users are also more eager to explore the individual verticals in future searches.

Since the use of combining verticals has been proven to be beneficial [5], the purpose of our research was to define and develop a model which when given a query decides whether to display the different verticals and integrating the model into a search engine. This is also called aggregated search [1]. The main research question of this paper is then: When and how should the different verticals

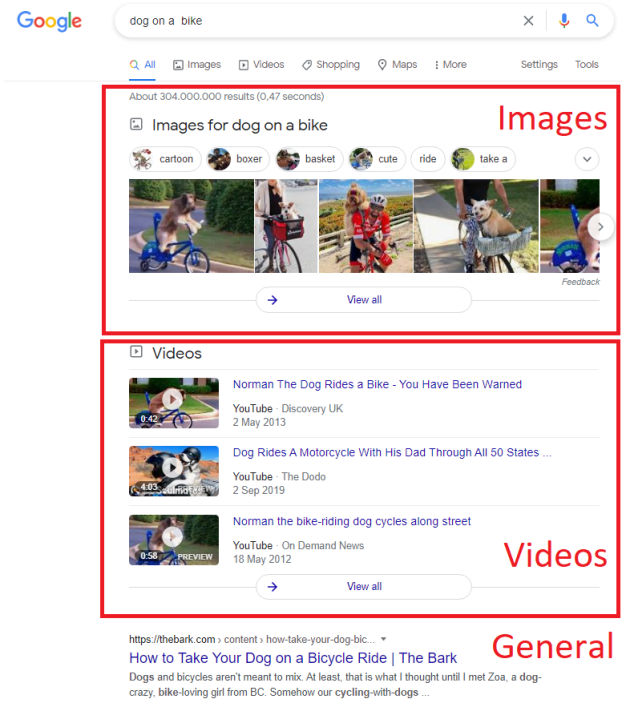

Figure 1: Heterogeneous Search Engine Result Page

be included? To tackle this question in a step-by-step manner the following research questions are answered:

(1) How can each vertical be identified using the content inside it?
(2) How to make a vertical selection for any query?
(3) How to display the selected verticals in a SERP?

We utilized the approach proposed in the thorough work of Arguello et al. [2] to answer the main research question. Firstly we proposed a way to collect documents for the corpus that is later used as training data. Website text TF-IDF and meta description TF-IDF features were extracted from the corpus of documents. The query synonym feature was extracted from the query itself. Features are further explained in the sections 2, 3, Using the features that were extracted from the corpus a classifier was trained. Finally, using the trained classifier and features extracted from the query, vertical selections were made for the evaluation queries. Our main findings are as follows: the generated vertical selections achieved an f-score of 0.4921 on the evaluation data set, this score shows that the research questions (1), (2) were answered, while the research question ’3 was not answered.

2 BACKGROUND

Heterogeneous search engine result pages describe a result page that contains one or more verticals. To elaborately decide whether to include such verticals or not, features need to be extracted from some information source. Then a model should be developed, in order to weigh those features and make a vertical selection.

2.1 Current Heterogeneous SERPs

Heterogeneous search engine result pages are ever evolving. This is because they belong in the field of information retrieval, which is constantly changing due to the technological advancements of search engines such as more thorough personalization of search results, change of content across the world wide web, different blocks that might need to be included, such as stock price history or live scores. There are various approaches to vertical selections, best of which [3] [1] combine different approaches to make one robust vertical selection model. Researchers came up with different metrics to optimize the generalizability and the robustness of the model. [15]

2.2 Types of Features

Arguello et al. [2] classified the features that are used to select verticals for heterogeneous search engine result pages into three categories: query features, vertical features, query-vertical features.

2.2.1 Query Features. Query features are query-specific because they only take the query itself as input. This means that they do not capture information that is specific to a vertical or the web. They are limited in effectiveness because of it. [2] They are, however, helpful in finding the users' primary goal when retrieving information, namely search intent, as found in the work of Tsur et al. [13] and Broder [4]. Search intent describes the motivation behind user's searches. Knowing the user's intent can help include certain verticals if needed. For example, if the query contains a question mark, the Q&A vertical might be relevant. The Q&A vertical contains content from websites that are focused on questions and answers. If the query has a dollar sign followed by some letters, the financial vertical which encapsulates websites that focus on stocks and other financial content might be relevant and so forth.

2.2.2 Vertical Features. Verticals features are features that consider the relation between verticals and the query. Previous search engine use data is needed to extract vertical features. An example of a vertical feature is the vertical-log feature, which tells how popular a certain vertical is. Knowing that certain verticals are more popular than others implies that they should be selected more often. The lack of usage data and the small amount of information they provide is the reason they rarely used in practice Arguello et al. [2].

2.2.3 Query-Vertical Features. The most widely used features are query-vertical features. [2] Since they consider the relationship between the individual verticals and the query. An example of such a feature is the vertical corpus feature. To extract the vertical corpus feature, first, a corpus of documents C must be collected. Each document D in the corpus C should be labelled with a subset of verticals V, that are relevant for the document. Once the corpus is collected, results returned by the query itself can be compared with the corpus to find the verticals that are fitting for the query. Once the corpus is collected, features from each document D can be extracted, the feature vectors then represent each vertical in the set V. These feature vectors can later be used to train a classifier.

2.2.4 ReDDE. Relevant Document Distribution Estimation Method (**ReDDE**) is a method for resource selection. Resource selection, as defined in the work of Nguyen et al. [9] aims at selecting different resources from which the results for a query should be selected. The resources in this research are the verticals. It samples a collection of documents from different resources into an index, after sampling, the index is queried and the resources that the top retrieved documents come from are deemed relevant for the query. ReDDE is a state-of-art approach that is heavily used today [1] [3] Since it uses the relationship between the query and the vertical, it is a subtype of the query-vertical feature.

2.3 Corpus Based Vertical Selection

Aggregated search has been studied extensively throughout the development of search engines. Various approaches that combine different types of features are present. Work of Shen et al. [12] used the vertical corpus feature in their work. Their approach was as follows: to classify queries into a predefined set of verticals, a corpus of preprocessed web pages was built. Labelled web pages were gathered using the Open Directory Portal[1]. The website content was retrieved and then preprocessed into a large collection of documents. The document counts in classes were equalized such that each class has a similar amount of documents. Document frequency and information gain were used for feature extraction. Statistical classifiers were trained using the features. The classifiers were combined to achieve better results. This is further supported by recent work of Mohandes et al. [8] which reviewed the techniques in which classifiers can be combined to achieve better performance. This is also called ensemble learning and helps increase the generalization of classifiers [16]. It was found that there is no one-fits-all way to combine them. An approach that is good in most cases can still perform much worse than another given the circumstances. It was, however, still concluded that fusing classifiers at the decision level improves performance. This means that each classifier predicts the class, and the predictions are combined using some rule e.g. weighting based on accuracy on the validation set. In the work of Shen et al. [12] the classifiers were fused at the decision level using weights that were calculated based on their accuracy on the training data set. The combination of classifiers was then used to classify preprocessed web pages returned by submitting a query to several popular search engines. If a query was short or ambiguous it was then enriched to retrieve more sensible results. The top-ranked verticals are deemed relevant to the query. Finally, a vertical selection can be made.

2.4 Vertical Selection Evaluation Criteria

As with most classification tasks, there is a possibility that a wrong vertical is selected. This damages the experience of the user. In order to give the best results, it should be minimized. Research of Zhou et al. [15] formulated 2 metrics for evaluating a vertical

[1] https://en.wikipedia.org/wiki/DMOZ

selection: **effectiveness** and **robustness**. Effectiveness measures how effective is a vertical selection across different types of users using the author proposed risk and reward functions. Robustness evaluates how robust is the vertical selection across all the users combined. After evaluating various methods, it was noticed that when only considering the reward function for the effectiveness metric, the majority of the approaches displayed similar results. However, after including the risk function, **ReDDE** and **CRCS** which are both vertical selection methods, achieved the best results. **CRCS** was also found to be the most robust method.

3 VERTICAL SELECTION METHODOLOGY

To make a vertical selection, verticals that will be selected are not must be defined. We selected the verticals that were proposed in the overview of the Federated Web Search Track data set of the Text Retrieval Conference of the year 2014 [6]. They were defined by manually classifying the resources that are were used for the resource retrieval task in the conference. The resources were classified into 24 different verticals, the verticals can be seen in appendix A.

3.1 Selected Features

From the three categories given in the section 2, two were selected. The features in the vertical feature category were not selected since they require previous use data. An example of use data is how often, compared to other verticals, is the content from one vertical selected. Such data is hard to acquire. Two types features that are used in this research are defined.

3.1.1 Query synonym feature. This feature gives a strong indication that the relevant must be included since keywords closely related to it are explicitly mentioned in the query. It can be argued that the keywords could be learned automatically; However, this requires a large data set that labelled queries with labels that are very similar to the selected verticals, due to time constraints in this research the feature will be extracted using a rule set, not a trained model.

3.1.2 Document text TF-IDF and meta descripton TF-IDF. The document text TF-IDF feature was successfully used in the work of Shen et al. [12] and it helps capture the type of terms that are popular among a collection of documents that represent a single vertical. The meta description TF-IDF is a feature that is extracted from the meta description of some document in the corpus. The meta description is a one-sentence description about the content of the website, it usually added by SEO managers or developers to help search engines capture the information in this website. An example of an extracted meta description is *One of the world's largest video sites, serving the best videos, funniest movies and clips.*. Websites that belong to the same vertical should have similar descriptions, therefore this feature should help identify the relevant verticals for a document. Both of these features require a corpus of documents to be retrieved and both are used to train the classifier.

3.2 Query Synonym Feature Extraction

To extract the query synonym feature, first, the rule set must be created. Similar to the definition given in the work of Arguello et al. [2], we defined rules as follows: a set of rules is: RS is a set of n rules $\{R_1, R_2, .., R_n\}$ each rule R_i maps words from many to one vertical as follows: $\{picture, image, pics\} \rightarrow images$. Given a set of vertical names $\{V_1, V_2, ..., V_n\}$ a rule is constructed in the following manner:

(1) Create a set RHS and add one vertical from the vertical list.
(2) Assign that set as the right-hand side of the rule.
(3) Using web dictionaries, such as Thesaurus[2], Meriam-Webster[3] and our intelligence come up with words $\{w_1, w_2, ..., w_k\}$ that are synonymous with or exclusive to the vertical in the set RHS and and them to the set LHS.
(4) Assign LHS as the left-hand side of the rule.

Then the rule is $\{w_1, w_2, ..., w_k\} \rightarrow V_1$, where w_i is a word that is synonymous with the vertical names. When a rule is applied to a word, if the word is in the set LHS, the vertical from the set RHS is relevant for that word.

Once the ruleset is created, given any query Q the feature vector can be calculated by dividing Q into a set of words $W = \{w_1, w_2, ..., w_i\}$, such as each word must be separated from other words by a white space character. Then for each word in the set W take the word and apply it to every rule in the set RS, union the results from all the rules and create a set of verticals. The verticals in the created set are considered relevant for the query. As an example take the query "black cat images", and apply the rule: $\{photo, image, pics\} \rightarrow Photo/Pictures$. Since the word image is in the query, the "Photo/Pictures" is relevant for the query. All the other verticals are not relevant since no other synonyms that belong to other rules were found. As seen from the example, the feature provides fast and accurate vertical suggestions, on the other hand, it can have wrong suggestions if a query like "image editor vacancies" is used.

3.3 Corpus features extraction

3.3.1 Corpus Retrieval. To tackle the research question (1), a labeled corpus of documents was collected. Research of Shen et al. [12] used website directories that categorize web pages to collect labelled documents, where the labels are the verticals the web page belongs to. At that time directories like these were a good source of information since they were constantly kept up to date and had a deep hierarchical categorization of web pages. After studying the most popular directory ODP, we noticed that the selected verticals do not always correspond with the categories provided. For example, when searching for categories that should contain web pages that belong to the social vertical, there are no categories like that found. Another example vertical is local, there is no such subcategory, and a similar category - regional in the ODP contains web pages that we would not assign to the local vertical. The web pages we found in ODP were sometimes outdated, originating from the '00s. If such web pages were to used as documents that represent a vertical, a vertical will be misrepresented, since the results returned by modern search engines will be different. This is because search engines try to retrieve the most relevant documents, and old websites are not mentioned often, since they are old. Such differences will decrease the accuracy of the classifier. Another approach for document collection is needed. Our proposed approach is to use web pages returned by querying the queries that explicitly target a

[2]https://www.thesaurus.com
[3]https://www.merriam-webster.com/thesaurus

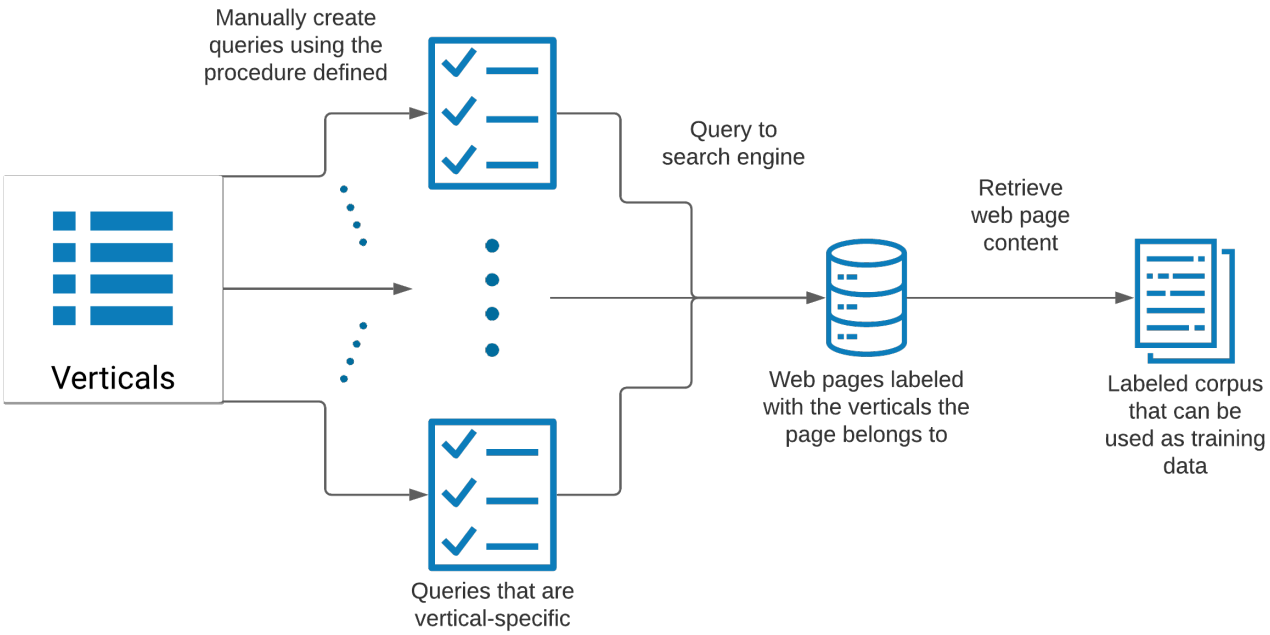

Figure 2: Corpus Retrieval Pipeline

single vertical to a web search engine as documents for the corpus. To select verticals for a query, the work of Shen et al. [12] compared the web pages returned by popular search engines for the query to the documents in the corpus. After we checked some of the results of querying vertical names and their synonyms, the websites returned are representative of that vertical. This categorization of web pages will not yield results as good as manually labelled by a collective of people, but can still yield good results as the categories are accurate in general. We visualized our proposed corpus retrieval pipeline, it can be seen in figure 2. Given a set of vertical names $\{V_1, V_2, ..., V_n\}$ a corpus can be collected as follows:

(1) For each vertical name V_i we generate a list of queries Q_i, the number of queries per vertical should be the same, to have a similar amount of documents per vertical.
(2) Then for each query in the list Q_i that belongs to the vertical V_i, use it to retrieve a set R of the top 100 results from some general web search engine.
(3) For each website in the set R, crawl the content of the website to obtain one document per website.
(4) Save the result as a document with label V_i to the corpus. If the website was no crawled successfully skip it and continue.

Using this approach a corpus containing labelled documents that will be used for further feature extraction and vertical selection can be obtained.

3.3.2 Website Text TF-IDF. Given a document D from the corpus, to extract the feature vector first the document must be stripped of all HTML tags. All of the punctuation, numbers are then removed. All of the remaining text is then switched to lower capital letters. For each word in the preprocessed document, its term frequency (**tf**) in the document is calculated and its inverse document frequency (**idf**) across all the documents is calculated. A feature vector of dimensions $(1 \times n)$ where n is the total number of words in all the documents is then extracted, where ith row of the vector will be the TF-IDF of the ith word.

3.3.3 Meta Description TF-IDF. It can be extracted similarly to the website text TF-IDF feature. Given a document D from the corpus, to extract the feature vector first the meta description must be extracted from the HTML of the document. This can be done by finding the `<meta>` tags in the document which have the property `name="description"`. Once the meta description is extracted, the TF-IDF feature vector is defined and extracted in the same way it was for the website text TF-IDF feature.

Both feature vectors are concatenated into one large feature vector of dimensions $(1 \times (n + m))$ where n is the total number of words in the website text of the documents and m is the number of words in the meta descriptions of the documents.

3.4 Query Classification

To answer the research question (2) we propose a method for query classification, in terms of verticals selected.

3.4.1 Classifier. The query classification pipeline is shown in the figure 3. First, the corpus features defined before are extracted for every document in the corpus and used to train a multi-class classifier. We use a multi-class classifier to make the implementation easier but using binary classifiers and the one-vs-rest approach

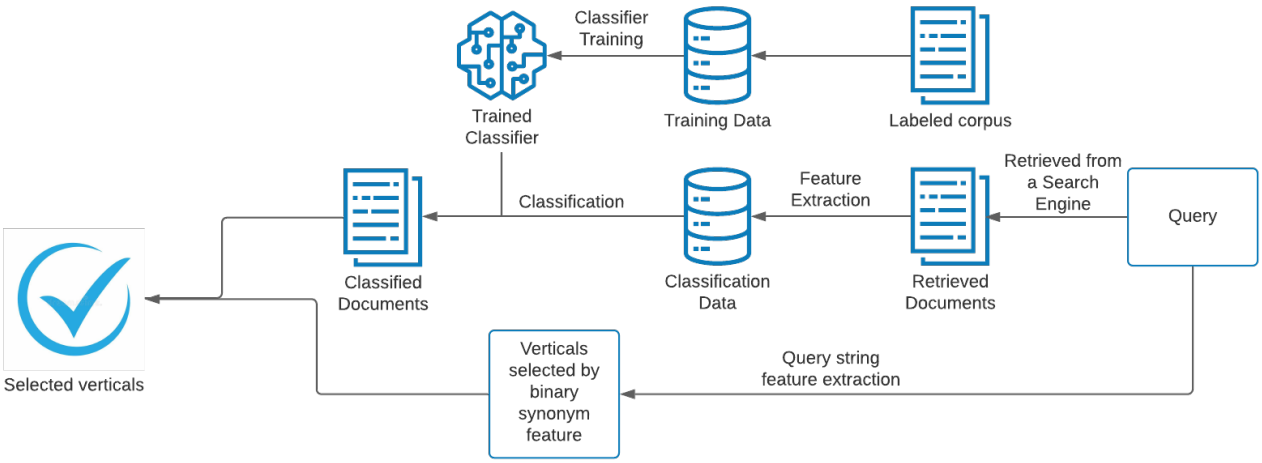

Figure 3: Query Classification Pipeline

is a viable alternative. In our proposed approach, it is assumed that the document belongs to one vertical, which is not always the case. Therefore, multi-label classifiers can be utilized to avoid this assumption and increase performance. We will evaluate the performance of XGBoost and the Random Forest classifiers. They were selected due to their performance in other text classification tasks. A comparative study of the performance of 4 classifiers was conducted in the work of Ramraj et al. [11]. Even though the text classification tasks differ from the ones tackled here, the study shows a general trend of the performance of different classifiers. The Random Forest classifier came out on top, while the XGBoost came in as the second best performing statistical classifier. Convolutional neural networks are not used due to time constraints and the fact that we are not familiar with them.

3.4.2 Classification. To classify a query using the corpus-vertical features, top 50 websites returned by a general web search engine are crawled in the same manner as documents were collected for the corpus. A query result is then a set QR which consists of n successfully retrieved documents. Each query document is then classified as belonging to one of the verticals using the trained classifier. Probabilities of the document belonging to verticals can also be used. This way the classifications are not binary, and the assumption that a document belongs to one vertical can be further removed.

Predicted verticals for each document are summed to obtain a vertical classification count. It is the number of times a document from the set QR is classified as belonging to some vertical. Verticals are included or excluded using a threshold t, in order to find the verticals that are popular among the results of the query. If the count is above a threshold t, the vertical is then selected for the query. Threshold t is a hyper-parameter that needs to be manually selected and can be used to optimize the vertical selection.

The relevant verticals selected by the query synonym feature are also retrieved and a union of both selections is used as the final result. A union is used because with both of the vertical selections we aimed to make the selections accurate, and not include verticals that *could* be relevant, rather including ones that *should* be relevant. Also, we did not find a more sensible way to combine the results.

3.5 Evaluation

Evaluation metrics used in this paper are as follows:

(1) Precision (**P**) - Number of correct verticals selected divided by the total number of verticals selected.
(2) Recall (**P**) - Number of correct verticals selected divided by the total number of correct verticals.
(3) F-Score (**F**) - Shows the overall accuracy, it is derived from precision and recall. Formula: $2 * \frac{P*R}{P+R}$

4 EXPERIMENTAL SETUP

4.1 Search Engine

The vertical selection was implemented using the back-end of SearchX [10]. SearchX is an open collaborative search engine that supports the execution of various collaborative search tasks. The back-end of SearchX uses BingApi to retrieve results, therefore BingApi is also used for corpus document retrieval.

4.2 Evaluation data set

To evaluate the vertical selection the 2014 Text REtrieval Conference (**TREC**) Federated Web Track data was used. Using this data set, the main findings of the research question (1) can be evaluated. The TREC 2014 consists of a set of 50 queries Q together with their labels. Queries can be seen in appendix B The Federated Web Track of the year 2014 includes three evaluation sets: resource selection, vertical selection, and results merging. For the evaluation of this paper, the vertical selection set is used. It is used because it is the most fitting for the vertical selection approach we propose. It contains a set of 24 vertical names V, which can be seen in appendix A. A text document that for each query in the set Q has either 0 or more relevant verticals from the set V that should be selected for that

Classifier	Threshold	Features used	P	R	F
Conference best performance[6]	N/A	documents	0.591	0.545	0.496
XGBoost	37%	website text TF-IDF, meta description TF-IDF, query synonym	0.57	0.4853	0.4814
RandomForest	37%	website text TF-IDF, meta description TF-IDF, query synonym	0.56	0.5003	0.4921
RandomForest	35%	website text TF-IDF, meta description TF-IDF, query synonym	0.54	0.5003	0.4788
RandomForest	40%	website text TF-IDF, meta description TF-IDF, query synonym	0.56	0.4953	0.4868
RandomForest	37%	website text TF-IDF	0.5400	0.4921	0.4801
RandomForest	37%	meta description TF-IDF	0.6100	0.4813	0.4967
N/A	N/A	query synonym	0.5500	0.4853	0.4734
RandomForest	10%	website text TF-IDF, meta description TF-IDF, query synonym	0.2727	0.5470	0.3243
Always predict General	N/A	N/A	0.72	0.4763	0.5407

Table 1: Classifier evaluation results

query is provided. Together with the evaluation set, an evaluation script is included, which calculates the precision, recall, and the f-score of the selection.

4.3 Query Synonym Feature Rule Set Creation

For the 24 verticals that are defined in the data set, a rule set consisting of 22 rules was created. It was created following the definition we gave in section 3.1.1, which was discussed in the work of Arguello et al. [2]. It contains 84 trigger words in total. For the 50 queries that are in the evaluation data set 5 were mapped to a vertical name, resulting in around 10% applicability rate. This means that the query synonym feature found relevant verticals in 10% of the queries, based on that feature alone. According to the ground truth provided by the TREC dataset, all but one of the the queries were classified correctly. The generated rule set can be found in the appendix D.

4.4 Corpus retrieval

Following the approach proposed in section 3.3.1, a query set for retrieving documents belonging to the different verticals was generated. These queries are used together with the vertical name, to retrieve documents that are specific to a vertical. The queries used can be found in appendix C. A corpus C with 3,747 documents was created. After extracting features, the size was reduced to 3,091 since not all documents contained a meta description, the percentage of documents that contain a meta description is then 83%. Each of the documents belongs to one vertical. On average, 134.4 documents belong to each vertical with a standard deviation of 9.6. The documents are the top-100 search results retrieved from the search engine using the queries, that were taken from the rule set, that was generated using the query string feature. The queries used for individual verticals can be seen in Appendix B. On average 44% of websites were successfully retrieved.

4.5 Query Documents

Using the method defined in the previous section, documents for each query in the TREC 2014 query set were collected. On average each query had 35.3 documents with a standard deviation of 2.5.

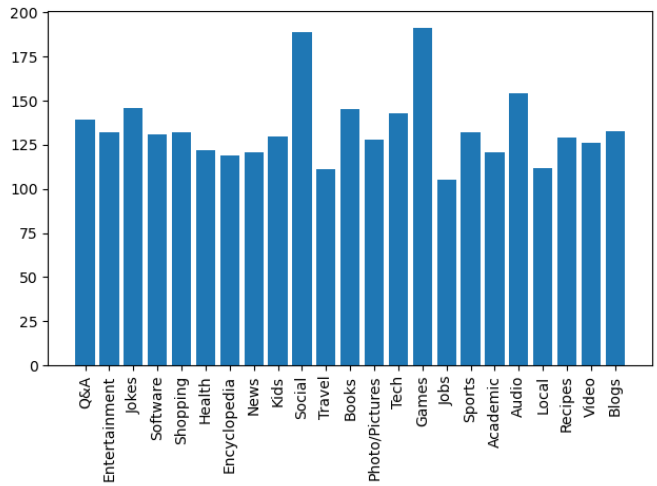

Figure 4: Corpus Document Distribution

5 RESULTS

5.1 Corpus Document Distribution

In figure 4 the distribution of documents among verticals can be seen. The Social and the Games verticals both have the largest document counts per class. The reason behind this is most likely the accessibility of the websites in those categories. Since the websites were retrieved using spoofed GET requests, they are not the same as regular requests from a browser, since browsers include various headers in the request which was not emulated while retrieving documents.

The verticals with the lowest document counts are Travel and Jokes verticals, the document counts in those classes is low because those are the two classes for which the requests timed out the most. Compared to the average of 134, those classes have 112 and 105 respectively, this is not that far away from the average, but since those classes are underrepresented in the training data, they could be classified incorrectly.

Once the corpus was retrieved and features extracted, the documents can then be displayed in 2D space. Using t-SNE which is

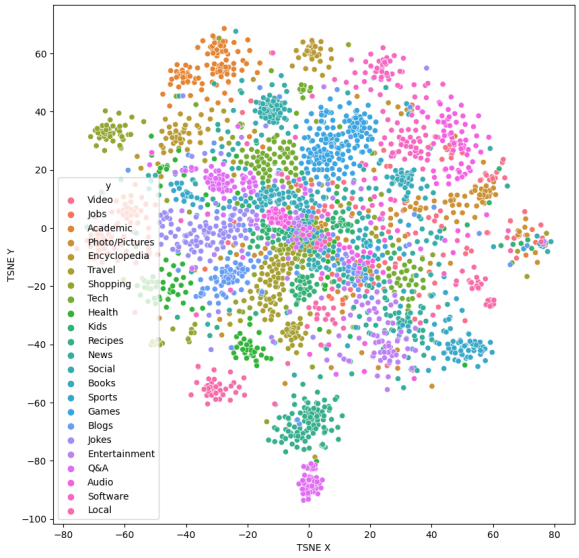

Figure 5: Corpus Document Class Distribution Visualization

a method of visualizing high dimensional data in 2-dimensional plots proposed in the work of van der Maaten and Hinton [14]. A plot of document class distribution that was generate using t-SNE can be seen in figure 5. From the plot, it can be seen that certain classes of documents are clustered, but there are a lot of points randomly positioned in the middle. From this plot, you can tell that the research question (1) was not tackled correctly. The differences between verticals are not captured correctly, since classes belonging to one document are scattered throughout the plot and not clustered together. In order to improve the segregation of vertical classes, either more features need to be extracted, or the documents in the corpus should be recollected using another approach. If the differences between verticals were captured more accurately, the different coloured dots should ideally be distributed uniformly throughout the plot and the different colours should remain in their individual clusters.

5.2 Overall Evaluation Results

To evaluate how well the research question (2) was answered, the vertical selection was evaluated using the evaluation dataset. In table 1 results of the different classifiers with different thresholds can be seen. We evaluated the Random Forest, XGBoost classifiers. Overall, the Random Forest classifier with a threshold of 37% gave the best results.

5.2.1 Threshold tuning. The threshold that can be seen in the results was chosen following the notion that no more than 2 verticals should be included per query. The selection should adhere to this rule since after the top 2 predictions the classifier almost always suggests a vertical that is not relevant for the query. This was tested using the evaluation data set, the lower the threshold the lower the precision of the classifications, while the recall, which indicates that the ratio of total correct answers found does not increase as much. If the threshold is set above 33.3% it impossible to select more than 2 verticals. This was done to find a threshold that gave the most sensible results. Such a process can lead to over-fitting and in general, should be avoided. It means that you fit your classifier based on the results of the evaluation data set, this will nearly always give you better results on that dataset, but in general, would reduce your results on unseen data. It was tuned in this case, because a fitting validation dataset was not found.

5.2.2 Score analysis. The score achieved using this classifier is very close to the maximum (0.496) that was achieved during the conference [6] which was a good result but might not reflect the true performance of the classifier since it was fitted on the dataset. After further investigation, it became clear that the vertical selection defined in this approach should receive high scores based on the ground truth results of the dataset. Since the *'General'* vertical is selected for every query, only the ranking of it is changed, this gives a lot of true positives and a small number of true negatives. As can be seen in the table 1, the recall of always predicting "General" is 0.47 which is almost half of the correct answers already. The best performing selection proposed in this research, which had the best f-score, only increases the recall by 2.5% which is around 3 correct classifications. If the threshold is further lowered to 10%, the achieved recall increases by 5% which is around 6 more correct classifications. Overall, the results of this vertical selection for this data set are only accurate when the verticals that need to be selected have a high count among the classified query results.

5.3 Ablation study

By excluding different features from the classifiers different effects on the score can be seen. This is also called an ablation study. As can be seen in the results table 1, scores only change by small amounts. Website text TF-IDF feature seems to be the most important one in classifying the queries. It has the highest recall of all the features. On the other hand, the query-synonym feature proves to be very accurate since, in theory, it should rarely classify false-positive results, as the keyword indicating that a vertical is relevant is mentioned in the query itself. Meta description seems to be the least important feature. It has the highest precision of all the individual features, but the recall increases by a minuscule amount. This indicates that most of the verticals are below the threshold, which means that the results are somewhat random and no majority was found. After a thorough analysis of the descriptions used for the classifier, it was found that only certain verticals can be identified accurately with the meta descriptions that are used since the descriptions often contain information that is not at all relevant to the vertical.

5.4 Individual Query Classification Analysis

In order to better understand the results of the classifier, results for three queries are analyzed. The results for the queries can be seen in table 2. The percentages seen near the vertical name in column 2, are the per cent of documents returned by the query that were classified as that vertical. If the percentage was below

Query	Results	Ground truth
row row row your boat lyrics	Kids 75%, General Audio 15% (excluded)	Video, General Blogs
punctuation guide	General, Academic 24% (excluded) Social 9% (excluded)	General, Blogs Encyclopedia Kids

Table 2: Results for individual queries

the threshold, which was set at 37%, the vertical was then excluded from the selection.

5.4.1 Query: row row row your boat lyrics. First query classification has 0% accuracy apart from the General vertical that we always include. After looking up the results returned by a modern search engine that did not have any information about the user, except for the location. The suggested verticals are music and videos, which makes sense since the query indicates that lyrics of some song are needed, meaning that videos of such songs are relevant, same for the music vertical. This shows that the ground truth is no longer correct, even though it was some time ago. No blogs showed up on the result page, an educated guess behind the discrepancies with the ground truth is that lyrics now have dedicated websites with a large collection of songs, blogs that contain such lyrics are therefore not as relevant for such queries. The "kids" vertical was selected following our approach for the query, it is not completely inaccurate since the song in the query is a popular kids song. This classification was most likely made due to documents in the corpus that are results of the query "kids songs". These documents are only labelled as "kids" since no "music" vertical is defined in the TREC 2014 verticals. Given this, it makes a lot of sense that the "kids" vertical is selected and indicates that the query terms used for corpus document retrieval should be selected more carefully.

5.4.2 Query: punctuation guide. The third query gave incorrect results. After looking at the results of a modern search engine using the same methods as it was done for the first query, most of the results are from university web pages that created extensive punctuation guides. Since the query set included the query "academic universities" for the vertical "academic", it makes sense that this classification was made, although it was not included for the query because the percentage is below the threshold.

6 RESPONSIBLE RESEARCH

All research should be conducted responsibly and take into account the different factors that might affect your research. The first part of this section will cover the reproducibility of this research, while the second part will cover the trustworthiness of the model that follows our proposed approach.

6.1 Reproducibility

Due to the approach suggested in this research, without the collected corpus the results will never be reproducible. This is because both the collection of corpus and the evaluation of the queries involves retrieving the top search results returned by BingApi. Search engines are always evolving, and the world wide web is always changing, so the content returned for one query at one time, might not be the same sometime later. To tackle this issue and make the research reproducible, the data used to obtain the results of this research can be downloaded *here*. Together with the data, the source code for training the model and generating visualizations can be found in a repository *here*. Instruction on how to run the evaluation are included in the `README.md` file that is in the repository.

6.2 Trustworthy Artificial Intelligence

Trust in AI systems as given in the work of Jacovi et al. [7] can be warranted and unwarranted. Trust in a system becomes unwarranted if it is possible to change the performance of the system, without affecting the trust. In this case, trusting that the model will provide the same correct results with a corpus that was collected manually, rather than using the one collected during the experimental phase of this research, would be unwarranted. Unwarranted trust should be avoided to prevent any misuses and false expectations. Since the method defined does not always lead to the same results, the trust in this model should be extrinsic. This means that the trust should come from observing the results provided by the model, and trusting it in situations in which it provided the correct results according to the user. Following these, unwarranted trust will be avoided and the model can be extrinsically trusted.

7 FUTURE WORK

Follow up work on this research should primarily focus on improving the corpus. Our proposed corpus collection approach was inspired by the work of Shen et al. [12] but modified due to the approach not being reproducible to capture all the needed verticals. To improve the corpus, the websites that are used should ideally be manually collected and labelled. This will solve two issues encountered in our approach. Since our documents were labelled automatically and were search engine results of queries that target a vertical, it is safe to assume that some web pages were mislabelled. The first issue that will be solved is then the fact that all documents will have the correct labels. Following the method we proposed to collect the corpus, an assumption was made that documents only belong to one vertical. This is not always the case for example both jokes and blogs vertical could be assigned to a document that was crawled from a blog website that is focused on jokes. Avoiding the false assumption would solve the second issue of documents belonging to a single vertical. By improving the corpus collection, the research question (1) would be tackled better, which in turn should improve the main findings of the research question (2).

REFERENCES

[1] Jaime Arguello. 2017. Aggregated Search. *Aggregated Search* XX, Xx (2017), 1–139. https://doi.org/10.1561/9781680832532

[2] Jaime Arguello, Fernando Diaz, Jamie Callan, and Jean-Francois Crespo. 2009. Sources of evidence for vertical selection. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. 315–322.

[3] Horatiu Bota, Ke Zhou, Joemon M. Jose, and Mounia Lalmas. 2014. Composite retrieval of heterogeneous web search. *WWW 2014 - Proceedings of the 23rd International Conference on World Wide Web* (2014), 119–129. https://doi.org/10.1145/2566486.2567985

[4] Andrei Broder. 2002. A taxonomy of web search. *ACM SIGIR Forum* 36, 2 (2002), 3–10. https://doi.org/10.1145/792550.792552

[5] Marc Bron, Frank Nack, and Lotte Belice Baltussen. 2013. Aggregated search interface preferences in multi-session search tasks. *SIGIR '13: Proceedings of*

the 36th international ACM SIGIR conference on Research and development in information retrieval 1 (2013).

[6] Thomas Demeester, Dolf Trieschnigg, Dong Nguyen, and Djoerd Hiemstra. 2013. Overview of the TREC 2014 Federated Web Search Track. *Proceedings of the Twenty-third Text REtrieval Conference, TREC-23* (2013), 1–12.

[7] Alon Jacovi, Ana Marasović, Tim Miller, and Yoav Goldberg. 2021. Formalizing trust in artificial intelligence: Prerequisites, causes and goals of human trust in AI. In *FAccT 2021 - Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. 624–635. https://doi.org/10.1145/3442188.3445923 arXiv:2010.07487

[8] Mohamed Mohandes, Mohamed Deriche, and Salihu O. Aliyu. 2018. Classifiers Combination Techniques: A Comprehensive Review. *IEEE Access* 6 (2018), 19626–19639. https://doi.org/10.1109/ACCESS.2018.2813079

[9] Dong Nguyen, Thomas Demeester, Dolf Trieschnigg, and Djoerd Hiemstra. 2016. Resource Selection for Federated Search on the Web. (2016). arXiv:1609.04556 http://arxiv.org/abs/1609.04556

[10] Sindunuraga Rikarno Putra, Felipe Moraes, and Claudia Hauff. 2018. SearchX: Empowering Collaborative Search Research.. In *SIGIR*. 1265–1268.

[11] S Ramraj, S Saranya, and K Yashwant. 2018. Comparative study of bagging, boosting and convolutional neural network for text classification. *Indian Journal of Public Health Research & Development* 9, 9 (2018), 1041–1047.

[12] Dou Shen, Rong Pan, Jian Tao Sun, Jeffrey Junfeng Pan, Kangheng Wu, Jie Yin, and Qiang Yang. 2006. Query enrichment for web-query classification. *ACM Transactions on Information Systems* 24, 3 (2006), 320–352. https://doi.org/10.1145/1165774.1165776

[13] Gilad Tsur, Yuval Pinter, Idan Szpektor, and David Carmel. 2016. Identifying web queries with question intent. *25th International World Wide Web Conference, WWW 2016* (2016), 783–793. https://doi.org/10.1145/2872427.2883058

[14] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html

[15] Ke Zhou, Ronan Cummins, Mounia Lalmas, and Joemon M. Jose. 2012. Evaluating reward and risk for vertical selection. *ACM International Conference Proceeding Series* (2012), 2631–2634. https://doi.org/10.1145/2396761.2398709

[16] Zhi-Hua Zhou. 2009. Ensemble Learning. *Encyclopedia of Biometrics* (2009), 270–273. https://doi.org/10.1007/978-0-387-73003-5_293

A VERTICALS USED FOR SELECTION

- General
- Video
- Jobs
- Academic
- Photo/Pictures
- Encyclopedia
- Travel
- Shopping
- Tech
- Health
- Kids
- Recipes
- News
- Social
- Books
- Sports
- Games
- Blogs
- Jokes
- Entertainment
- Q&A
- Audio
- Software
- Local

B QUERIES IN THE EVALUATION DATASET

- Vera Pavlova
- Asian culture
- the raven
- Alek Wek
- hermitian conjugate
- reinforcement learning
- holiday houses ardennes
- song of ice and fire
- Natural Parks America
- awk trim non-printable characters
- price gibson howard roberts custom
- Adam rogers music
- falsifying pictures 21st centurie
- Ezz-thetic
- How much was a gallon of gas during depression
- grimm episodes
- what is the starting salary for a recruiter
- raleigh bike
- Cat movies
- why do leaves fall
- ted talk mooc
- dodge caliber
- vice president residence
- pita recipe
- aluminium extrusion
- fabric glue
- severed spinal cord
- seal team 6
- weather in nyc
- blink reflex
- constitution of italy
- hobcaw barony
- contraceptive diaphragm
- uss stennis
- turkey leftover recipes
- earthquake
- punctuation guide
- mud pumps
- squamous cell carcinoma
- salmonella
- who was lincolns vice president
- route 666
- council bluffs
- 347 ford engines
- silicone roof coatings
- lomustine
- roundabout safety
- flight simulators
- hague convention
- largest alligator on record
- collagen vascular disease
- welch corgi
- iowa girls high school basketball
- elvish language
- hospital acquired pneumonia
- grassland plants
- detroit riot
- basil recipe
- assumption of mary
- row row row your boat lyrics
- what causes itchy feet
- carmen electra video
- causes of the cold war
- cayenne pepper plants
- volcanoe eruption
- reduce acne redness
- little johnny
- alkan piano
- navalni trial

- barcelona real madrid goal messi
- running shoes boston
- kobe bryant news
- board games teenagers
- convert wav mp3 program
- criquet miler

C QUERIES USED FOR CORPUS RETRIEVAL

```
{
  "Video": [
    "videos",
    "video clips",
    "youtube videos"
  ],
  "Jobs": [
    "jobs hiring",
    "job vacancy",
    "internships"
  ],
  "Academic": [
    "university academic",
    "academic research",
    "academic paper"
  ],
  "Photo/Pictures": [
    "photos",
    "images",
    "pictures"
  ],
  "Encyclopedia": [
    "wiki",
    "encyclopedias online",
    "encyclopedia definition"
  ],
  "Travel": [
    "flights",
    "travel vacation",
    "travel sightseeing"
  ],
  "Shopping": [
    "shopping online",
    "e-commerce order",
    "buy online"
  ],
  "Tech": [
    "tech news",
    "tech advancements",
    "tech giants"
  ],
  "Health": [
    "Healthiness",
    "bodybuilding",
    "health injury"
  ],
  "Kids": [
    "children toys",
    "Kids cartoons",
    "kids songs"
  ],
  "Recipes": [
    "recipes",
    "recipe ingredients",
    "chicken recipes"
  ],
  "News": [
    "recent news",
    "news broadcast",
    "newspapers"
  ],
  "Social": [
    "social media",
    "facebook",
    "insta",
    "tweets"
  ],
  "Books": [
    "novel books",
    "kindle",
    "top books"
  ],
  "Sports": [
    "sport news",
    "goal",
    "nba"
  ],
  "Games": [
    "first person shooter games",
    "best table top games",
    "video games"
  ],
  "Blogs": [
    "blog",
    "blog diary",
    "blog sites"
  ],
  "Jokes": [
    "stand-up jokes",
    "comedian",
    "funny anecdote"
  ],
  "Entertainment": [
    "celebrities tv",
    "party news",
    "entertainment"
  ],
  "Q&A": [
    "q&a",
    "how to boil egg",
    "What is the meaning of life"
  ],
  "Audio": [
    "mp3 audio download",
    "movie soundtracks",
```

```
    "audio"
  ],
  "Software": [
    "software install",
    "software apps",
    "server"
  ],
  "Local": [
    "local weather",
    "restaurants near me",
    "local election"
  ]
}
```

D GENERATED QUERY SYNONYM RULE SET

```
{
  "General": [],
  "Video": [
    "video",
    "clip",
    "vid",
    "youtube"
  ],
  "Jobs": [
    "hiring",
    "job",
    "vacancy",
    "salary",
    "linkedin",
    "internship"
  ],
  "Academic": [
    "university",
    "college",
    "research",
    "education"
  ],
  "Photo/Pictures": [
    "photo",
    "image",
    "pics",
    "photograph"
  ],
  "Encyclopedia": [
    "wiki",
    "encyclopedia",
    "definition"
  ],
  "Travel": [
    "flights",
    "vacation",
    "sight",
    "travel"
  ],
  "Shopping": [
    "order",
    "commerce",
    "shopping"
  ],
  "Tech": [
    "tech"
  ],
  "Health": [
    "fitness",
    "bodybuilding",
    "injury",
    "pain",
    "ache",
    "Health"
  ],
  "Kids": [
    "children",
    "cartoon",
    "kids",
    "toy",
    "baby"
  ],
  "Recipes": [
    "recipe",
    "ingredients"
  ],
  "News": [
    "news",
    "broadcast",
    "newspaper",
    "paparazzi"
  ],
  "Social": [
    "social",
    "facebook",
    "insta",
    "twitter",
    "tweet"
  ],
  "Books": [
    "novel",
    "kindle",
    "book"
  ],
  "Sports": [
    "sport"
  ],
  "Games": [
    "fps",
    "mmo",
    "gaming",
    "third-person",
    "xbox",
    "playstation"
  ],
  "Blogs": [
```

```
      "blog"
    ],
    "Jokes": [
      "stand-up",
      "comedian",
      "joke",
      "humour",
      "anecdote"
    ],
    "Entertainment": [
    ],
    "Q&A": [
      "why",
      "how",
      "when",
      "where",
      "?"
    ],
    "Audio": [
      "mp3",
      "flac",
      "wav",
      "soundtrack",
      "audio"
    ],
    "Software": [
      "install",
      "app",
      "server",
      "software"
    ],
    "Local": [
      "weather",
      "near",
      "forecast",
      "local"
    ]
  }
```