M.Sc. Thesis

Relevant snippet extraction in biomedical question answering

Maria-Varvara Georgiou
EY1308

Supervisor: Ion Androutsopoulos
Assistant Supervisor: Prodromos Malakasiotis

ATHENS, MAY 2015
Acknowledgements

I would like to thank my supervisor, Ion Androutsopoulos, for his consistent and restless guidance throughout this thesis and generally through this MSc program, as well as Professor Theodore Kalaboukis, for his advice and comments. I would also like to thank Makis Malakosiotis, who co-supervised my work, for his advice, the discussions, support and patience during the creation of this thesis. A big thanks goes to my family, Sofia, Tasos and Dimitris, who stood by me all those years and made everything possible.
1 Introduction

1.1 Goal of the thesis

Experts in the biomedical sciences search for answers in the literature on a daily basis. Most of them use PubMed, which is a search engine that allows keyword queries to be entered and returns a list of possibly relevant documents (mostly biomedical journal articles), as illustrated in Figure 1. However, the experts usually need answers to specific questions, ideally formulated as natural language questions, something that typical search engines like PubMed do not support yet. Question Answering (QA) systems [17, 12, 4] and biomedical QA systems in particular [2, 3] could be useful in such cases, since they combine Information Retrieval (IR) with Natural Language Processing (NLP) techniques in order to automatically answer natural language questions. Figure 2, shows the architecture of a typical QA system.

![Figure 1: A real-life biomedical question from BioASQ, used as a search query in PubMed.](http://www.ncbi.nlm.nih.gov/pubmed)
Figure 2: A typical QA system architecture. The dotted box indicates the focus of this thesis, snippet extraction.

The focus of this thesis is Snippet Extraction. For each question we assume that all the truly relevant documents are given (i.e., that a perfect Information Retrieval component is available) and we aim to identify the snippets of the given documents that are the most relevant to the question (as illustrated in Figure 3).
1.2 The BioASQ competition

Biomedical knowledge is dispersed in hundreds of heterogeneous knowledge sources and it is growing rapidly (Figure 4). This causes difficulty to the biomedical experts, who are in constant need of highly specialized information during their research. The BioASQ competition [16] pushes for a solution to this information access problem by setting up a challenge on Biomedical Semantic Indexing and Question Answering (QA).

2http://www.bioasq.org/
BioASQ comprises two main tasks:

- **Task A: Large–Scale Online Biomedical Semantic Indexing.** In this task the participants are asked to tag new PubMed documents with classes from the MeSH hierarchy before they are tagged (classified) by biomedical curators manually.³

- **Task B: Biomedical Semantic QA.** In this task, the participants are given English questions constructed by biomedical experts. The participating systems are required to respond with relevant concepts, articles (from designated article repositories), relevant snippets (from the relevant articles), relevant RDF triples (from designated ontologies), ‘exact’ answers (e.g., named entities in the case of factoid questions) and ‘ideal’ answers (paragraph-sized summaries). This thesis focuses on Task B of BioASQ and more specifically on the part that aims to identify relevant snippets.

³https://www.nlm.nih.gov/mesh/
1.3 Overview of the thesis

The remainder of this thesis is organized as follows. Chapter 2 describes the system that was developed during the thesis. Chapter 3 presents the datasets that were used in the thesis and the experimental results. Chapter 4 discusses previous related work. Finally, Chapter 5 concludes and proposes ideas for further improvements.
2 The system of this thesis

Given a question $Q$ and the set of gold relevant documents $D$ of $Q$ (documents selected as relevant to $Q$ by human experts), our system attempts to detect and return the snippets (e.g., sentences) of $D$ that are relevant to $Q$. We treated the problem as a binary classification problem. Specifically, for each candidate snippet of a relevant document, a classifier decides whether the snippet is relevant to $Q$ or not. In an initial set of experiments, the candidate snippets were individual sentences. In later experiments, the candidate snippets were three consecutive sentences each, as discussed further below.

2.1 Tools

2.1.1 MetaMap

MetaMap [1] is a tool developed at the National Library of Medicine (NLM) to map biomedical text to the UMLS Metathesaurus.\footnote{http://metamap.nlm.nih.gov/} The latter is a large, multi-purpose, and multi-lingual thesaurus that contains millions of biomedical and health related concepts, their synonyms, and their relationships.\footnote{http://www.nlm.nih.gov/pubs/factsheets/umlsmeta.html} MetaMap takes free text as input and analyzes it, in order to identify concepts of UMLS mentioned in the text.
Phrase: heart attack
Meta Candidates (Total=6; Excluded=0; Pruned=0; Remaining=6)
1000 C0027051: Heart Attack (Myocardial Infarction) [Disease or Syndrome]
861 C0018787: HEART (Heart) [Body Part, Organ, or Organ Component]
861 C0277793: Attack (Onset of illness) [Temporal Concept]
861 C1261512: attack (Attack behavior) [Social Behavior]
861 C1281570: Heart (Entire heart) [Body Part, Organ, or Organ Component]
861 C1304680: Attack (Observation of attack) [Finding]
Meta Mapping (1000):
1000 C0027051: Heart Attack (Myocardial Infarction) [Disease or Syndrome]

Figure 5: Example of MetaMap output.

Figure 5 shows an example output of MetaMap for the phrase “heart attack”. MetaMap has returned five possible candidates for this phrase. The candidates are sorted according to the MetaMap score, which indicates the relevance of a UMLS concept to the tagged phrase. The candidate with the highest score is the one which is finally mapped to the phrase. Figure 6 explains further the details of each candidate that MetaMap returns in the output.

Figure 6: Explanation of MetaMap output.
For the purposes of this thesis all the questions and documents were tagged with MetaMap. The MetaMap output parts that we use to compute features used by the classifier are the following:

- **Concept Id**: The unique Id of the concept.

- **Preferred name**: The name of the concept (which may be the same with the matched string).

- **Matched Words**: A list representing (1) the correspondence of words in the candidate concept and words in the original phrase, and (2) any lexical variation in the matching.

- **Semantic types**: A list of categories the identified concept belongs in (e.g., heart attack has semantic type “Disease or Syndrome”).

### 2.1.2 Word2Vec

Word2vec [9][10][11] is a tool which takes a text corpus as input and maps its words to continuous dense space vectors, so that, roughly speaking, syntactically or semantically related words are mapped to nearby vectors. The resulting vectors can be used as features in many natural language processing and machine learning applications. In this thesis, we used the word vectors that were provided by BioASQ. To construct these vectors, Word2vec was applied to a corpus of 10,876,004 English abstracts of biomedical articles from PubMed. The resulting vectors of 1,701,632 distinct words (types) are publicly available from [http://bioasq.lip6.fr/tools/BioASQword2vec/](http://bioasq.lip6.fr/tools/BioASQword2vec/).

---

6[https://code.google.com/p/word2vec/](https://code.google.com/p/word2vec/)
2.2 System architecture

![Diagram of system architecture](Diagram.png)

Figure 7: System overview.

To decide whether a sentence is a relevant snippet or not, we follow a pipeline approach. First we preprocess the (relevant to the question) documents. Each document is split into sentences with the Stanford Tokenizer [8] and tagged with MetaMap.\(^7\) For every token in each sentence, the corresponding continuous space vector is also found. Using the continuous space vectors of the words and the MetaMap tags, the sentences are then transformed into feature vectors, which are then classified by a Logistic Regression classifier as corresponding to relevant snippets or not.\(^8\)

2.3 Features

A first version of the system used only bag-of-words (BoW) features. We represented each question and each sentence of a relevant document, as a vector which contained the TF-IDF scores of their tokens. We then computed the cosine similarity between the vector of the question and each one of the sentences in order to learn a threshold which would distinguish the relevant snippets from the irrelevant ones. We also used the cosine similarity as a feature in an initial logistic regression classifier, along with token-level features such as edit distance between the question and each one of the sentences, common words between the question and the sentences etc. However the results were poor (recall \(\approx 2\%\), precision \(\approx 2\%\)) and therefore we decided to follow a conceptual

\(^7\)http://nlp.stanford.edu/software/tokenizer.shtml

\(^8\)We use the LIBLINEAR implementation of logistic regression; consult http://www.csie.ntu.edu.tw/~cjlin/liblinear/.
retrieval model [18], based on MetaMap concepts and continuous space vectors constructed by Word2Vec. An example of an expert’s question with a relevant snippet, which was also marked by the expert, taken from BioASQ’s competition data is the following:

**Question:** “Is c-myc subject to regulation by the circadian clock?”

** Relevant Snippet:** “Loss of Bmal1 reduced the expression of per1, per2, per3, wee1 and p53. The expression of p21 and c-myc was also altered in certain cell lines.”

One can see that there are very few common words between the question and the snippet, and also there is no obvious lexical relevance. We also note that the common token (“c-myc”), which seems to be an important indicator that the snippet is relevant, is repeated very often in the particular document, and also in many non-relevant snippets. Hence, treating the sentences that contain it as relevant snippets would produce too many false positives.

### 2.3.1 MetaMap features

Given a question $Q$ and a sentence $S$, we compute the following features based on MetaMap. The reader is reminded that given two sets $S_1$ and $S_2$ their Dice coefficient is:

$$\frac{2 \cdot |S_1 \cap S_2|}{|S_1| + |S_2|}$$

1. Dice coefficient of common distinct MetaMap concept IDs between $Q$ and $S$.
2. Dice coefficient of common distinct MetaMap preferred names between $Q$ and $S$.
3. Dice coefficient of common distinct MetaMap matched words between $Q$ and $S$.
4. Dice coefficient of common distinct MetaMap semantic types between $Q$ and $S$.
5. Common distinct semantic types between $Q$ and $S$, divided by the number of the distinct semantics types of $Q$; intuitively, this is the recall of $Q$’s semantic types in $S$.
6. Common distinct concept IDs between $Q$ and $S$ divided by the number of distinct concept IDs of $Q$; intuitively, this is the recall of $Q$’s concept IDs in $S$.
7. The same features are also computed between $Q$ and the previous sentence of $S$ and between $Q$ and the next sentence of $S$. 

12
2.3.2 Continuous space vectors features

Given a question $Q$ and a sentence $S$, we compute the following features based on continuous space word vectors computed by Word2Vec (Section 2.1.2).

1. **Euclidean distance based similarity using centroids.** This is computed as:

   $$Sim(Q, S) = \frac{1}{1 + d(c(Q), c(S))} \tag{2}$$

   where $c(Q), c(S)$ are the centroid vectors of $Q$ and $S$ respectively, defined below and $d(a, b)$ is the Euclidean distance between two vectors $a, b$ which is computed as follows:

   $$d(a, b) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \tag{3}$$

   In order to compute the centroid of a sentence $S$ (or a question $Q$), $S$ (or $Q$) is first tokenized and for each token $t$ the corresponding continuous space vector $v(t)$ is retrieved. The centroid of $S$ (similarly, $Q$) is then computed as follows, where $n$ is the number of tokens of $S$ (or $Q$).

   In effect, the centroid also takes into account the term frequencies of the tokens.

   $$c(S) = \frac{1}{n} \sum_{i=1}^{n} v(token_i) \tag{4}$$

2. **Euclidean distance based similarity using weighted centroids.** This is the same score as the one of Equation 2, with the difference that the vector of each token is weighted by the corresponding inverse token frequency ($idf$) score. Hence, the centroid of $S$ (similarly, $Q$) is now computed as follows:

   $$c(S) = \frac{\sum_{i=1}^{n} v(token_i) \cdot idf_i}{\sum_{i=1}^{n} idf_i} \tag{5}$$

   where the $idf$ scores have been computed on 10,876,004 documents of PubMed as follows:

   $$idf(token_i) = \log\left(\frac{10,876,004}{f_{token_i}}\right) \tag{6}$$

   and $f_{token_i}$ is the frequency of each distinct token in the collection.
3. **Euclidean distance based similarity between the sentence and question tokens.**

To compute this set of features, we first create a set containing the tokens of $S$ and a set containing the tokens of $Q$. We then compute the similarities between all the possible pairs of tokens $(t_q, t_s)$ between the two sets as follows:

$$
Sim(t_q, t_s) = \frac{1}{1 + d(v(t_q), v(t_s))} \cdot \frac{idf_{t_q} \cdot idf_{t_s}}{\text{maxIdf}^2}
$$

(7)

where $v(t_q)$ and $v(t_s)$ are the continuous space vectors for a question and a sentence token, respectively, and $\text{maxIdf}$ is the maximum $idf$ of all the distinct tokens of the training data, that we used to train our final system.

We then compute the following features:

- Maximum of the similarity scores $Sim(t_q, t_s)$, over all of the pairs of tokens $t_q$, $t_s$ of the question and sentence, respectively.
- Minimum of all the similarity scores $Sim(t_q, t_s)$.
- Median of all the similarity scores $Sim(t_q, t_s)$.
- Average of the three minimum similarity scores $Sim(t_q, t_s)$.
- Average of the three maximum similarity scores $Sim(t_q, t_s)$.
- Average of all the similarity scores $Sim(t_q, t_s)$.

4. The same features as in 3, but without weighting with $idf$ scores. Hence, the new similarity is computed as follows:

$$
Sim(t_q, t_s) = \frac{1}{1 + d(v(t_q), v(t_s))}
$$

(8)

where $v(t_q)$ and $v(t_s)$ are again the continuous space vectors for a question and a sentence token, respectively.
3 Experiments

3.1 Dataset

3.1.1 Raw data

For the purposes of this thesis we used the benchmark datasets released by BioASQ during the challenges of 2013 and 2014. These datasets contain 782 questions along with their gold (annotated by biomedical experts) relevant documents and snippets. The questions are of four categories:

- **Yes/no Questions:** The “exact” answer of such a question is either “yes” or “no”. For instance, “Are there any DNMT3 proteins present in plants?” is a yes/no question.

- **Factoid Questions:** The “exact” answer of such a question is a named entity (e.g., a protein). For instance, “What is the methyl donor of DNA (cytosine-5)-methyltransferases?” is a factoid question.

- **List Questions:** The “exact” answer of such a question is a list of named entities. For instance, “Which species may be used for the biotechnological production of itaconic acid?” is a list question.

- **Summary Questions:** Such a question can be answered only with a paragraph-sized summary (i.e., these questions do not have “exact” answers). For instance, “How do histone methyltransferases cause histone modification?” is a summary question.

The snippets marked by the experts, may have one the following forms:

- A full sentence, starting with an uppercase character and ending with a period (or question mark etc). In the 2014 challenge of BioASQ, snippets had this form.

- A part of a sentence, or a sequence of sentences where the first and the last sentence is not necessarily a full sentence. In the 2013 challenge of BioASQ, snippets had this form.

Below, we show an example of a question along with its relevant documents and snippets (provided by the experts).

- **Question:** “Mutation of which gene is associated with Achondroplasia?”

- **Relevant Documents**
Relevant snippets:

- “She was subsequently diagnosed with hypochondroplasia at the age of 6 years when disproportional short stature, stocky habitus and macrocephaly were observed. These phenotypic findings were later confirmed by the presence of fibroblast growth factor receptor 3 (FGFR3) gene mutation N540K.” [http://www.ncbi.nlm.nih.gov/pubmed/21225389]

- “Achondroplasia can be diagnosed by characteristic clinical and radiographic findings in most affected individuals. In individuals who may be too young to diagnose with certainty or in individuals with atypical findings, molecular genetic testing can be used to detect a mutation in FGFR3.” [http://www.ncbi.nlm.nih.gov/pubmed/20301331”]

- “Achondroplasia comes from the genetic point mutations in the fibroblastic growth factor receptor 3 gene (FGFR3), which enables abnormal cartilage growth-plate differentiation and insufficient bony development. The most common genetic mutations in this receptor are G to A at position 1138 (G1138A), which result in the substitution of glycine to arginine at codon 380.” [http://www.ncbi.nlm.nih.gov/pubmed/15221641]

- “To investigate the mutation at the transmembrane domain of fibroblast growth factor receptor 3 (FGFR3) nucleotide 1138 site for identifying the major pathologic mechanism of achondroplasia (ACH) and to evaluate the efficacy of denaturing gradient gel electrophoresis (DGGE) method for screening the point mutations.” [http://www.ncbi.nlm.nih.gov/pubmed/12048679]

- “Nucleotide 1138 in transmembrane domain of FGFR3 gene is the hot point for mutation in ACH and hence its major pathologic cause.” [http://www.ncbi.nlm.nih.gov/pubmed/12048679]
- “The nucleotide 1138 of FGFR3 gene is also the hotspot of mutation in Chinese patients with ACH. A simple and rapid molecular diagnostic method has been set up to differentiate ACH from other similar genetic dwarfism.”

- “The diagnosis of achondroplasia was based on the presence of clinical and radiographic findings and confirmed by the presence of a common FGFR3 gene mutation (Gly380Arg) detected by restriction enzyme analysis and sequencing of the polymerase chain reaction products.”

- “All patients carried the same glycine-to-arginine mutation at codon 380(G380R) of the transmembrane domain of FGFR3.”

- “The results suggest that the G380R mutation of FGFR3 is also a frequent mutation causing achondroplasia in Chinese. Detecting FGFR 3 gene mutation with PCR-SSCP and restriction enzymes analysis is a convenient, rapid and reliable molecular diagnostic assay for prenatal and early diagnosis of achondroplasia.”

- “To evaluate whether mutation in the exon 10 of the fibroblast growth factor receptor 3(FGFR3) gene in common in Chinese patients with achondroplasia.”

- “Achondroplasia (ACH) is the most frequent form of short-limb dwarfism. Recently, the gene mutation responsible for ACH has been identified in the transmembrane domain of the fibroblast growth factor receptor 3 gene. The cause of ACH is a point mutation at nucleotide 1138 of the cDNA, resulting in the substitution of an arginine residue for a glycine.”

- “Recent studies mapped the achondroplasia gene on chromosome region 4p16.3 and identified a common mutation in the gene encoding the fibroblast growth factor receptor 3 (FGFR3).”
3.1.2 Data preprocessing

In order to be easier to process, all the relevant documents and snippets were split into sentences with the sentence splitter of the coreNLP library [8]. The snippets that the biomedical experts had marked were then ‘rounded’ to the nearest beginning and end of a sentence. If a ‘rounded’ snippet contained many sentences, we treated each of these sentences as a different relevant snippet, as illustrated in Fig. 8.

![What the expert marked as snippet:]

....the remaining 2 fetuses although echogenic findings persisted into the third trimester. In a low-risk population, echogenic bowel usually resolves without neonatal sequelae. Even when persistent into the third trimester, echogenic bowel does not uniformly herald an abnormal outcome. Echogenic bowel coexistent with other abnormalities (such as growth deficiency or structural malformations) may be a comarker for aneuploidy.

![What we consider as snippets:]

....the remaining 2 fetuses although echogenic findings persisted into the third trimester. In a low-risk population, echogenic bowel usually resolves without neonatal sequelae. **Even when persistent into the third trimester, echogenic bowel does not uniformly herald an abnormal outcome. Echogenic bowel coexistent with other abnormalities (such as growth deficiency or structural malformations) may be a comarker for aneuploidy.**

Figure 8: The initial snippet marked by the expert is ‘rounded’ to entire sentences and then split into two different snippets.

Figures 10, 11 and 12 show the distribution of the resulting sentence-long snippets over the classes (relevant, irrelevant) in the training, development and test sets we constructed. One can see that there is a class imbalance problem (almost 90% of the snippets are not relevant).

---

9 http://nlp.stanford.edu/software/corenlp.shtml
Figure 9: Distribution of relevant and irrelevant sentence-long snippets in the training data.

Figure 10: Distribution of relevant and irrelevant sentence-long snippets in the development data.
To alleviate the class imbalance problem, in later experiments we used a sliding window definition [15] of relevant and irrelevant snippets. Each window was three sentences long and it was considered a relevant snippet if it contained at least one (rounded) sentence marked by the human expert as relevant. In the example of Figure 9, windows x, y, z are considered relevant, while windows a, b, c are irrelevant snippets. The human expert had marked as relevant sentences S1 and S7.

Figure 11: Distribution of relevant and irrelevant sentence-long snippets in the test data.

Figure 12: Using a sliding window for snippet representation. The human expert had marked the sentences S1 and S7 as relevant. Windows x, y, z are now considered relevant, while windows a, b, c are considered irrelevant snippets.
In Figures 13, 14 and 15 one can see the class distribution with the sliding window definition of snippets. The class imbalance has improved, though the majority of the windows are again irrelevant to the corresponding questions.

Figure 13: Distribution of relevant and irrelevant windows in the training data.

Figure 14: Distribution of relevant and irrelevant windows in the development data.
3.2 Evaluation measures

To evaluate the system we used the following measures.

\[
Recall = \frac{TP}{TP + FN} \tag{9}
\]

\[
Precision = \frac{TP}{TP + FP} \tag{10}
\]

\[
F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{11}
\]

Where \(TP\) are the snippets (single sentences or windows of three sentences, depending on the experiment) which our system correctly classified as relevant (true positives), \(TN\) are the snippets that the system correctly classified as irrelevant (true negatives), \(FP\) are the snippets which were misclassified as relevant (false positives), and \(FN\) are the snippets that were misclassified as irrelevant (false negatives).

3.3 Experimental results on the development data

All the experiments were performed twice, with the single-sentence snippets and window-snippets. The plots in the following sections show the precision - recall scores of the system on development
data, trained on different percentages of the training data, while changing the confidence threshold of the logistic regression classifier.

### 3.3.1 Single-sentence snippet results on development data

Figure 16 illustrates the results of the system that uses all the features of Section 2.3. We observe that using more than 40% of the training data does not improve the performance of the classifier, i.e., adding more data would not improve the performance of our system.

![Figure 16: Precision-recall curves of the classifier, using single-sentence snippets and varying the size of the training dataset.](image)

The next experiment (Figure 17) concerns the same features, but now applying Principal Component Analysis (PCA) to the data. We kept only the 30 eigenvectors with the highest eigenvalues, since the eigenvalues of the other three eigenvectors were close to zero. We observe that PCA allowed the system to reach higher precision scores, but for very low recall. Overall, PCA did not lead to any improvement in practice.
Figure 17: Precision-recall scores with PCA, using single-sentence snippets and varying the size of the training dataset.

Figure 18 shows the contribution of the two kinds of features, i.e., MetaMap features (Section 2.3.1) and features based on continuous space word vectors (Section 2.3.2). In all cases, the classifier is trained on the entire (100%) training dataset. Without the MetaMap features, i.e., when using only the Word2Vec-based features, the performance of the classifier is very poor; most of the data points are close to zero precision and recall, which is why the curve has a different shape. Hence, the MetaMap features are crucial. Nevertheless, when the Word2Vec features are added, the performance of the classifier improves significantly, i.e., both kinds of features are useful.
Figure 18: Contribution of the two kinds of features, using single sentence snippets and the entire training set.

3.3.2 Sliding windows system results on development data

Figure 19 illustrates the results of the system which uses all the features of Section 2.3 apart from the features of bullet 7 in Section 2.3.1. We removed the latter features because there is no previous or next sentence now, but previous and next window. Since the overlap with the previous and the next window is large, the use of these features seemed unnecessary. So far this version of the system is the best. This can be explained by the fact that the class imbalance is now smaller (Fig. 13-15) and the fact that we now ask the system to identify a wider area (a three-sentence window, rather than a particular sentence) that contains a sentence marked by a human expert as relevant. Figure 19 shows that it is possible to obtain precision and recall scores both in the 60 - 70% region at the same time.
The experiment of Figure 20 concerns the same features with Figure 19, but applying PCA to the data. We kept only the 18 eigenvectors with the highest eigenvalues, since the eigenvalues of the other eigenvectors were close to zero. As in Section 3.3.1, PCA allowed the classifier to reach higher precision scores (now comparing Fig. 20 to Fig. 19), but with lower recall scores. Hence, again PCA did not lead to any improvement in practice.

Figure 21 shows the contribution of the two kinds of features, i.e., MetaMap features (Section 2.3.1 apart from bullet 7) and features based on Word2Vec (Section 2.3.2). In all cases, the classifier is trained on the entire (100%) training dataset. Again, without the MetaMap features the perfor-
Performance of the classifier is very poor; most of the data points are close to zero precision and recall, which is why the corresponding (blue) curve has a different shape. Hence, the MetaMap features are again crucial. Nevertheless, when the Word2Vec features are added to the MetaMap features, the performance of the classifier improves significantly, i.e., both kinds of features contribute.

Figure 21: Contribution of the two types of features involved, when using the sliding window snippets and the entire training set.

3.4 Experimental results on the test data

In order to further evaluate our system, we used the test data (Fig. 11) in additional experiments. Apart from precision-recall curves, for these experiments, we also show curves for Mean Interpolated Precision (MIP) at different recall levels, and Mean Average Interpolated Precision (MAIP) scores [7], defined as follows:

- **Mean Interpolated Precision (MIP):** For each question \( Q_k \) of the test dataset, we rank each snippet \( S_i \) by descending confidence (of the classifier) that it is a relevant snippet. We then mark each \( S_i \) in the list with its true category (relevant or irrelevant snippet). Afterwards we compute the precision \( P_{Q_k}(S_i) \) and the recall \( R_{Q_k}(S_i) \) up to each relevant \( S_i \) (including the relevant \( S_i \)) in the list as follows:

\[
P_{Q_k}(S_i) = \frac{\text{relevant snippets of } Q_k \text{ up to } S_i}{i}
\]

\[
R_{Q_k}(S_i) = \frac{\text{relevant snippets of } Q_k \text{ up to } S_i}{|R_k|}
\]
where $R_k$ is the set of relevant snippets for question $Q_k$. For each recall level $r \in \{0.0, 0.1, 0.2, \cdots, 1.0\}$, we then compute that Interpolated Precision $IP_{Q_k}(r)$ of question $Q_k$ as follows:

$$IP_{Q_k}(r) = \max_{i: R_{Q_k}(S_i) \geq r} P_{Q_k}(S_i)$$  \hspace{1cm} (14)

By averaging over all the questions $Q_k$, we obtain the Mean Interpolated Precision at recall level $r$:

$$MIP(r) = \frac{1}{K} \sum_{k=1}^{K} IP_{Q_k}(r)$$  \hspace{1cm} (15)

where $K$ is the number of test questions.

- **Mean Average Interpolated Precision (MAIP):** To compute the MAIP score of a system, we first average its $IP_{Q_k}(r)$ over all recall levels, separately for each question $Q_k$, obtaining its Average Interpolated Precision for $Q_k$:

$$AIP_{Q_k} = \frac{1}{11} \sum_{r \in \{0.0, \ldots, 1.0\}} IP_{Q_k}(r)$$  \hspace{1cm} (16)

By averaging over all the questions, we obtain the MAIP score:

$$MAIP_{Q_k} = \frac{1}{K} \sum_{k=1}^{K} AIP_{Q_k}$$  \hspace{1cm} (17)

### 3.4.1 Single-sentence snippet results on test data

Figure 22 shows the precision-recall curves of the system on test data, trained on different percentages of the training data, while changing the confidence threshold of the logistic regression classifier. The results are slightly worse than the ones in the corresponding experiments on the development data (Fig.16).
Figure 22: Precision-recall curves on the test dataset using single-sentence snippets and varying the size of the training dataset.

Figure 23 shows the Mean Interpolated Precision $MIP(r)$ at different recall levels using different percentages of the training data. The overall Mean Average Interpolated Precision score when all the training data are used is $MAIP = 0.3316$. Table 1 shows the $MAIP$ scores as a function of the percentage of training data used.

Figure 23: Mean interpolated precision ($MIP(r)$) at different recall levels, measured on the test data, using single-sentence snippets and varying the size of the training dataset.
<table>
<thead>
<tr>
<th>Training data (%)</th>
<th>MAIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20%</td>
<td>0.3110</td>
</tr>
<tr>
<td>40%</td>
<td>0.3225</td>
</tr>
<tr>
<td>60%</td>
<td>0.3285</td>
</tr>
<tr>
<td>80%</td>
<td>0.3277</td>
</tr>
<tr>
<td>100%</td>
<td>0.3316</td>
</tr>
</tbody>
</table>

Table 1: Mean Average Interpolated Precision (MAIP) scores on test data, when using different percentages of the training data, with single-sentence snippets.

### 3.4.2 Sliding windows snippet results on test data

Figure 24 shows the precision-recall curves of the system on test data, trained on different percentages of the training data, while changing the confidence threshold of the logistic regression classifier. The results are very similar to those obtained on the development data (Fig. 19).

![Figure 24: Precision-recall curves on the test dataset using sliding-window snippets and varying the size of the training dataset.](image)

Figure 24: Precision-recall curves on the test dataset using sliding-window snippets and varying the size of the training dataset.

Figure 25 shows the Mean Interpolated Precision (MIP($r$)) at different recall levels, using different percentages of the training data. The overall Mean Average Interpolated Precision score when all the training data are used is $MAIP = 0.6667$. Table 2 shows the MAIP scores as a function of the percentage of training data used. Clearly, increasing the size of the training dataset would not (on its own) help our system perform better.

![Figure 25: Mean Interpolated Precision (MIP($r$)) at different recall levels.](image)
Figure 25: Mean interpolated precision ($MIP(r)$) at different recall levels, computed on the test dataset, using sliding-window snippets and varying the size of the training dataset.

Table 2: Mean Average Interpolated Precision (MAIP) scores on the test dataset, when using different percentages of the training data, with sliding-window snippets.
4 Related work

Passage retrieval [15, 6] is the task of retrieving only parts (e.g., paragraphs) of documents which are relevant to a user’s natural language question (e.g., “Which genes have been found mutated in Gray platelet syndrome patients?”) or keyword query to a search engine (e.g., “mutated genes” AND “Gray platelet syndrome”). It is very useful for focusing on specific parts of documents which seem to have the most useful information with respect to the user’s question or query.

In the biomedical domain, one of the most motivating reasons is the exponential growth of the biomedical literature over time as shown by Hunter and Cohen [5] (see also Figure 4). A system that would not only retrieve relevant documents, but would also highlight the most relevant parts of them could be very useful to biomedical scientists during their research. For example, it could help them figure out more quickly if a document is relevant to their information needs, and identify more easily the parts that provide the information they search for.

In classic passage retrieval, many methods have been proposed. The standard approaches are based on the identification of important sentences. Usually each sentence is assigned a score (or weight) depending on its perceived importance according to the user’s query (or question) in relation to the text under consideration. The score (or weight) for each sentence is computed mostly based on bag-of-words like characteristics of the sentence and the question (or query). Such characteristics are common tokens between the question (or query) and the sentence, part-of-speech tags, tf-idf scores of each token etc. [14] Subsequently, the most important sentences are grouped and processed for particular purposes (e.g. summarization).

Unfortunately the approaches described above do not perform well in the biomedical domain, since the user’s queries (or questions) often have very few common terms with the relevant passages (see also Section 2.3). In order to address this problem, a popular technique for biomedical passage retrieval is to recognize biomedical entities or concepts both in the question and the documents and then detect possible relations between the question and the candidate passages based on this information. Zhou et al. [18] proposed a system which first identifies concepts in the question (in this case concepts are entry terms from MeSH gene symbols from the Entrez gene database). Then for each paragraph of a document \(d\), the system searches for the subset of sentences which contain all the concepts of the query and in the highest density possible. According to their results this approach brings significant improvements in comparison to the best known methods.
Finally, an issue which concerns the researchers of this area is determining the optimum size for a passage in order to be easier to be retrieved and also satisfy the needs of the user. Salton et al. [15] studied three different variations of passages: paragraph sized passages, where a whole paragraph is considered to be a passage, dynamic length passages, where the length of the passage depends on the length of the question and the length of the document, and “sliding window” passages, where the potential passages are extracted using a sliding window of constant length of $n$ sentences (Section 3.1.2). Although the sliding window had the second best results, and dynamic length had slightly better results, the conclusion was that the sliding window is better, because dynamic length requires much effort to be tuned for each document.
5 Conclusions and future work

During this thesis a system was developed which given a biomedical question in natural language and its relevant documents (journal articles), extracts relevant (to the question) snippets from the relevant documents. The system was trained on data (questions, relevant snippets) of the BioASQ competition. It uses a logistic regression classifier with features based on mostly MetaMap tags and continuous space word vectors (produced by Word2Vec). Bag-of-word features worked very poorly, since there is often no word overlap between the question and its relevant snippets.

The experimental results indicate that it is difficult for the system to extract single-sentence snippets. We found out that it is easier for the system to extract three-sentence windows, aiming to have at least one relevant sentence per extracted window. In the latter scenario, the experimental results show that it is possible to obtain precision and recall scores both in the 60 - 70% region, with a Mean Average Interpolated Precision ($MAIP$) of 0.667. By contrast, with single-sentence snippets the $MAIP$ score was 0.331.

Future work could consider using continuous space word vectors of higher dimensionality (e.g., 500 - 600 dimensions compared to the 200 dimensions we used in this thesis), and word vectors produced by GloVe [13], which is an alternative to using Word2Vec.\footnote{http://nlp.stanford.edu/projects/glove/} \footnote{https://github.com/akkineniramesh/pubmedcentral-glove} Another possibility would be to construct word vectors using a preprocessed corpus, where words matched by MetaMap would have been replaced by MetaMap concepts (and similarly for questions and snippets), or to experiment with Deep Learning methods (e.g., convolutional neural networks).\footnote{http://cs224d.stanford.edu/}
References


