11-797: Question Answering
Final Report

Team: The Flying Riddlers

Vasu Sharma, Nitish Kulkarni, Srividya Pranavi Potharaju, Gabriel Bayomi
[vasus, nitishkk, spothara, gbk, ehn, teruko] @cs.cmu.edu
Language Technologies Institute
Carnegie Mellon University

May 9, 2018

1 Introduction

In the last few years, there have been tremendous advancements in the field of Question Answering, with the neural Question Answering systems outperforming even humans in some cases [12]. However, there is also a large diversity in the type and nature of existing QA systems, and most systems are very specific to a certain dataset. For a new QA dataset, it is still hard to predict what kind of QA models would be most befitting and the extent to which a QA model trained on one dataset would work on another. To that end, we present a comprehensive investigation into different types of QA systems based on question and answer types, as well as carefully analyze the relative merits and demerits of having a modularized question-type based system as opposed to an end-to-end QA system.

1.1 General Hypothesis

With the key idea to assess the generalizability of a Question Answering system, we formulate our general hypothesis as:

For the task of Question Answering (QA), a comprehensive and holistic system can be designed that can handle multiple question/answer types and can yield a good performance across a number of QA datasets.

1.2 Scope

In order to test our hypothesis, we design and implement broadly two types of Question Answering systems - 1) a modularized question-type specific system and 2) an end-to-end system. We test the performance of these systems on two different datasets (one small and one large) in order to identify a general pattern of relative performances.

We confine the scope of this work to the question types - yes/no, factoid, list, summary and description. We also consider fixed-sized manually-curated datasets in order to build supervised QA systems. The key focus of our work is, as mentioned earlier, to compare the relative performances of the two systems across different datasets and it is not to develop a single system that works best on a specific dataset. That being said, we do employ several techniques to improve the individual performances of both the systems in order to ensure a fair comparison.

2 Relevant Literature

The Question Answering problem is immensely popular among NLP researchers and large industrial research labs all over the world. The release of the SquAD data set [18] boosted this interest as many researchers began to use this data set to evaluate their approaches. However, the SquAD data set is only used to evaluate extractive question-answering systems and hence doesn't allow for free-form answers. The state of the art model for this problem presently is the Dynamic Co-Attention Network (DCN) from Xiong et al. [28], the ensemble model of which presently holds the top spot on the SquAD leader board. The DCN first fuses co-dependent representations of the question and the document in order to focus on relevant parts of both. Then a dynamic pointing decoder iterates over potential answer spans. This iterative procedure enables the model to recover from initial local maxima corresponding to incorrect answers.

Other prominent works include the R-Net networks from Microsoft Research Asia [11] who introduce self matching networks to tackle this problem. They first match the question and passage with gated attention-based recurrent networks to obtain the question-aware passage representation. Then they propose a self-matching attention mechanism to refine the representation by matching the passage against itself, which effectively encodes information from the whole passage. Finally they employ the pointer networks to locate the positions of answers from the passages.

The Bidaf network from Seo et al [21] was one of the first landmark papers to demonstrate the effectiveness of attention based bidirectional recurrent networks on the task of Machine Comprehension. They introduced a multi-stage hierarchical process that represents the context at different levels of granularity and uses bi-directional attention flow mechanism to obtain a query-aware context representation without early summarization. Many subsequent works on the problem now build on top of the Bidaf network.[10] from CMU and [22] for Microsoft Research Redmond are some of the other prominent works on this problem. In [10] the authors encode the information from syntactic trees into the vector embeddings and use this added structured syntactic knowledge to improve their results on this task. In [22], the authors propose a reinforcement learning based approach to determine how many passes should the network make over the knowledge base during answer generation thereby doing away with restriction of fixed number of passes over the knowledge base.

Biomedical Question answering has always been a hot topic of research among the QA community at large due to the relative significance of the problem and the challenge of dealing with a non standard vocabulary and vast knowledge sources. The BioASQ challenge has seen large scale participation from research groups across the world. One of the most prominent among such works is from Chandu et al. khyati-paper who experiment with different biomedical ontologies, agglomerative clustering, Maximum Marginal Relevance (MMR) and sentence compression. However, they only address the ideal answer generation with their model. Peng et al. fudan in their BioASQ submission use a 3 step pipeline for generating the exact answers for the various question types. The first step is question analysis where they subdivide each question type into finer categories and classify each question into these subcategories using a rule based system. They then perform candidate answer generation using POS taggers and use a word frequency-based approach to rank the candidate entities. Wiese et al. fastqa propose a neural QA based approach to answer the factoid and list type questions where they use FastQA: a machine comprehension based model [26] and pre-train it on the SquaD dataset [18] and then finetune it on the BioASQ dataset. They report state of the art results on the Factoid and List type questions on the BioASQ dataset. Another prominent work is from Sarrouti and Alaoui usmba who handle the generation of the exact answer type questions. They use a sentiment analysis based approach to answer the yes/no type questions making use of SentiWordNet for the same. For the factoid and list type questions they use UMLS metathesaurus and term frequency metric for extracting the exact answers.

Despite the popularity of Machine Comprehension problems, very little work has been done on free-form answer generation for questions. The biggest reasons likely include the lack of large standard data sets, difficulty in evaluation and the inherent difficulty of ‘human-like’ natural language generation. One of the only existing data sets for this problem is the MS Marco dataset released by Microsoft [12]. The top positions on this leader board are held by modifications of the RNet [11] and Reasonet [22] algorithms. Most of these approaches generate the answer by using an attention based bidirectional LSTM network to generate the answer word-by-word conditioned on the knowledge base, question and the best answer span. However, most of these answers lack syntactic coherence, retention of long term dependencies and human-like answer quality, which have been the primary problems with free-form natural language generation. Here we explore approaches to try and solve these problems and try to improve the performance on Machine Comprehension and free-form response generation.

3 Dataset

We build our QA systems on two datasets: BioASQ dataset (from PubMed documents) and MAchine Reading COmprehension (MS MARCO) dataset (from web documents).

3.1 BioASQ

BioASQ dataset was released by the BioASQ challenge [1], which is a large scale biomedical question answering and semantic indexing challenge that has been running as an annual competition since 2013. This challenge assesses the ability of systems to semantically index very large numbers of biomedical scientific articles, and to return concise and user-understandable answers to given natural language questions by combining information from biomedical articles and ontologies. We deal with the Phase B of the challenge which deals with large scale biomedical question answering. The dataset provides a set of questions and snippets from PubMed, which are relevant to the specific question. It also provides users with a question type and urls of the relevant PubMed articles itself. The 5b version of this dataset consists of 1,799 questions in 3 distinct categories:

1. **Factoid type**: This question type has a single entity as the ground truth answer and expects the systems to output a set of entities ordered by relevance; systems are evaluated using the mean reciprocal rank [17] of the answer entities with reference to the ground truth answer entity.

2. **List type**: This answer type expects the system to return an unordered list of entities as answer and evaluates them using a F-score based metric against a list of reference answer entities which can vary in number.

3. **Yes/No type**: This question type asks the systems to answer a given question with a binary output namely yes or no. The questions typically require reasoning and inference over the evidence snippets to be able to answer the questions correctly.

The dataset expected the participants to generate two types of answers, namely, exact and ideal answers. In ideal answers, the systems are expected to generate a well formed paragraph for each of the question types which explains the answer to the question. They call these answers 'ideal' because it is what a human would expect as an answer by a peer biomedical scientist. In the exact answers the systems are expected to generate "yes" or "no" in the case of yes/no questions, named entities in the case of factoid questions and list of named entities in the case of list questions.

3.2 MS MARCO

MS MARCO dataset is intended for non-commercial research purposes only to promote advancement in the field of artificial intelligence and related areas. In MS MARCO, all questions are sampled from real anonymized user queries. The context passages, from which answers in the dataset are derived, are extracted from real web documents using the most advanced version of the Bing search engine.The answers to the queries are human generated. Finally, a subset of these queries has multiple answers.Compared to previous publicly available datasets, this dataset is unique in the sense that

- all questions are real user queries
- the context passages, which answers are derived from, are extracted from real web documents
- all the answers to the queries are human generated
- a subset of these queries has multiple answers
- all queries are tagged with segment information

The complexity of the query varies from category to category The v1 dev version of this dataset consists of questions whose answer type can be broadly categorized into 5 distinct categories using a classifier which is trained on human labeled data:

- NUMERIC: This category constitutes of 28.14% of the dev data and expects a numeric value as its answer. E.g.: 'xbox release data'
- ENTITY: This category constitutes of 10.5% of the dev data and expects a named entity as its answer.
- LOCATION: This category constitutes of 5.12% of the dev data and expects a location/ place as its answer
- PERSON: This category constitutes of 2.4% of the dev data and expects a person name as an answer
- DESCRIPTION (Phrase): This is the majority category of the dev data comprising of 53.8%. The description could vary from one single line to any number of lines (no limit). E.g.: How to cook a turkey?

4 Question-type based QA system

We now present a QA system which is a combination of several modules, each built to handle a specific question type. The reason we believe that such a system would be effective is because of that, in this case, the nuances of each of the question types can be appropriately captured in the modules and the modifications to one module to address its corresponding inadequacies will not affect the performance of any other. We build this system for the BioASQ dataset, primarily because of the unique challenges that the BioASQ dataset which call for a modularized system.

In the sections that follow, we shall present the modules for handling the question types: yes/no, factoid/list and summary type. In addition to addressing each question type specifically, we also cater to different answer types: exact and ideal. Exact answers represent the subset of the BioASQ task where the responses are not structured paragraphs, but instead either a single entity (*yes/no* types) or a combination of named entities (*factoid* or *list* types) that compose the correct reply to the given query. The main idea refers to evaluating if a response is able to capture the most important components of an answer. For factoid or list types of questions, we must return a list of the most likely entities to compose the answer. The main difference between them is that ground truth for *factoid* questions is composed of only one correct answer and the evaluation method is Mean Reciprocal Rank (MRR). However, the ground truth for *list* is an actual list of correct answers with varying length, which uses F-measure as an evaluation metric. The BioASQ submission format allows everyone to submit 5 ranked answers for *factoid* and 1 to 10 answers for *list.* For *yes/no* questions, the ground truth is simply the yes or no label, using F-measure as an evaluation metric.

4.1 Yes/No Type Questions

Although yes/no questions require a simple binary response, calculating yes/no responses for a BioASQ question can be challenging for the following reasons:

1. There is an inherent class-bias towards the questions answered by `yes` in the dataset
2. The dataset is quite small for training a complex semantic classifier
3. An effective model must perform reasoning and inference using the limited information it has available, which is extremely difficult even for non-expert humans

Due to the nature of the question type, these questions can not be simply classified by using word-level features. Learning the semantic relationship between the question and the sentences in the documents is quite elemental to solving this task. Hence, we adopt a Natural Language Inference (NLI)-based system that learns if the assertions made by the questions are true in the context of the documents. Since most of the well-performing NLI models today are neural models, one of the biggest obstacles for this approach would be to be able to augment the BioASQ dataset with existing NLI datasets in a meaningful fashion.

4.1.1 Hypotheses

Two key challenges that we faced in building a yes/no classifier were - a) the shallow bag-of-words models cannot capture the semantic relationships that are required to answer a yes/no question, and b) the are inadequate NLI datasets for biomedical domain, and the large NLI datasets that are available employ significantly different vocabularies using Global Vectors for Word Representation (GloVe) embeddings.

In light of these challenges, we devise the following hypotheses for the yes/no type questions:

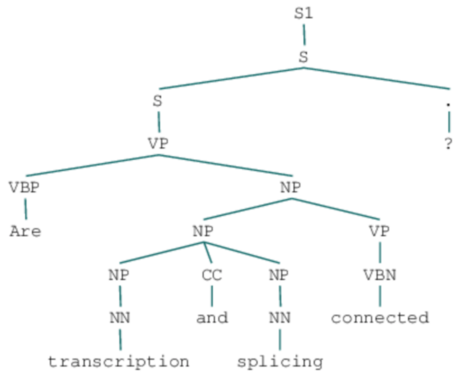

Figure 1: The parse tree of an example question as generated by the BLLIP parser

1. Yes/No questions can be modeled as an NLI task by creating assertions from the questions and assessing textual entailment among the snippets/documents

2. We can enable transfer learning for BioASQ dataset using pre-trained neural models that use GloVe embeddings by projecting bio embeddings as well as GloVe embeddings to a common space

To test these hypothesis, we train a Recognizing Textual Entailment (RTE) model for a standard NLI dataset, and fine-tune it for the BioASQ dataset using our proposed word-embedding projection technique. We then generate assertions from questions and evaluate the entailment or contradiction of these assertions using the RTE model. Using these entailment scores for all the sentences in the snippets or documents, we heuristically evaluate the answer to the yes/no question.

4.1.2 Assertion Extraction

The first step towards answering the question is to identify the assertions made by the question. For this, we use a statistical natural language parser to identify the syntactical structure in the question. We, then, heuristically generate assertions from the questions.

Consider the following example question:

Is the monoclonal antibody Trastuzumab (Herceptin) of potential use in the treatment of prostate cancer?

Upon parsing of this question, we have the phase constituents of the question. As shown in the example in Figure 1, almost all yes/no questions have a standard format that begins with an auxiliary verb followed by a noun phrase. In this example, we can toggle the question word with the first noun phrase to generate the assertion:

The monoclonal antibody Trastuzumab (Herceptin) is of potential use in the treatment of prostate cancer.

In a similar manner, we then create positive assertions for all *yes/no* questions as depicted in Table 1. As a simple extension to this, we can also create negative assertions by using *not* along with the auxiliary verbs.

Question	Assertion
Is the protein Papilin secreted?	The protein Papilin *is* secreted
Are long non coding RNAs spliced?	long non coding RNAs *are* spliced
Are transcription and splicing connected?	Transcription and splicing *are* connected.
Is RANKL secreted from the cells?	RANKL *is* secreted from the cells.
Does metformin interfere thyroxine absorption?	Metformin *does* interfere thyroxine absorption.
Has Denosumab (Prolia) been approved by FDA?	Denosumab (Prolia) *has* been approved by FDA.

Table 1: Assertion generation for some questions from training set in BioASQ Phase 6b by heuristic-based rearrangement of the auxiliary verb the questions starts with.

4.1.3 Recognizing Textual Entailment

The primary goal of our NLI module is to infer if any of the sentences among the answer snippets entails or contradicts the assertion posed by the question. We segment the answer snippets for each question to produce a set of assertion-sentence pairs. To then evaluate if these assertions can be inferred or refuted from the sentences, we build a Recognizing Textual Entailment (RTE) model using the *InferSent* model [6], which computes sentence embeddings for every sentence and has been shown to work well on NLI tasks. In training *InferSent*, we experience two major challenges:

1. The number of assertion-sentence pairs in BioASQ is too few to train the textual entailment model effectively.

2. The models that are pre-trained on SNLI [2] datasets use GLOVE [15] embeddings that cannot be used for biomedical corpora which have quite different characteristics and vocabulary compared to the corpora that GLOVE was trained on.

However, we have pre-trained embeddings available that were trained on PubMed and PMC texts along with Wikipedia articles [16]. To leverage these embeddings, we implement an embedding-transformation methodology to projecting the PubMed embeddings to GLOVE embedding space and then fine tune the pre-trained *InferSent* on the BioASQ dataset for textual entailment. The hypothesis is that, since both the embeddings had a significant fraction of documents in common (Wikipedia corpus), by transforming the embeddings from one space to another, the sentence embeddings from the model would still represent a lot of the semantic features of the input sentences that can subsequently used for classifying textual entailment. For this task, we explore both linear and non-linear methods of embedding transformation.

Linear transformation

While simple, a linear projection of embeddings from one space to another has shown to be quite effective for a lot of multi-domain tasks. By imposing an orthogonality constraint on the project matrix, we model this problem as an orthogonal Procrustes problem:

Let d_p and d_g be the embedding dimensions of PubMed embeddings and GLOVE embeddings respectively. If E_p and E_g are the matrices of PubMed embeddings ($N \times d_p$) and their corresponding GLOVE embeddings ($N \times d_g$) for the words that both the embeddings have in common (N), the projection matrix ($d_g \times d_p$) can be computed as,

$$W^* = \arg\min_W \| W E_p^\intercal - E_g^\intercal \|$$

subject to the constraint that W is orthogonal.

The solution to this optimization problem is given by using the singular value decomposition of $E_g^\intercal E_p$, i.e.

$$W^* = UV^\intercal,$$

$$\text{where,}$$

$$E_g^\intercal E_p = U\Sigma V^\intercal$$

With this simple linear transformation, we then compute the transformed embeddings for all the words in the PubMed embeddings that are not present in the GLOVE embeddings.

Non-Linear Transformation

We also explore a non-linear transformation using a feed-forward neural network, the objective is to learn function f such that,

$$f(e_p; \theta) = e_g$$

where, e_p and e_g are PubMed and GLOVE embeddings respectively. We model f using a deep neural network with parameters θ, and train using the common words in both the embeddings. The effectiveness of this training is a result of the large number of common vocabulary between the two embeddings (since both are trained on Wikipedia text among other corpora).

The transformed embeddings from these models were used in conjunction with the pre-trained *InferSent* model to encode the semantic features of the biomedical sentences as sentence embeddings. Subsequently, we employ these sentence embeddings of the assertion-sentence pairs for a particular question to train a three-way neural classifier to predict if the relationship between the two is entailment, contradiction or neither. It is worth noting here that the embedding transformation techniques that we implemented are not specific to the NLI tasks and, in fact, enable transfer learning of a much broader set of tasks on smaller datasets like BioASQ by using the pre-trained models on large datasets of other domains and fine-tuning on the smaller dataset.

4.1.4 Classification

As a final step, we use the textual entailment results for each assertion-sentence pair generated to heuristically classify the answer as *yes* or *no.* Since our system comprises multiple stages with the errors of each cascading to the final stage, we do not get perfect entailment results for the pairs. However, since we have a lot of pairs, we aggregate these entailment scores to compute the overall entailment or contraction scores to reduce the effect of accumulated errors for individual pairs on classification.

We use a simple unsupervised approach for classification by just comparing the overall entailment and contradiction scores, i.e. if the total number of snippet sentences that entail the assertion are N_e and the total number of snippet sentences that contradict are N_c, then,

$$\text{answer}_\text{q} = \begin{cases} \text{yes} & \text{if } N_e \geq N_c \\ \text{no} & \text{otherwise} \end{cases}$$

The end-to-end architecture of our system from the input questions and snippets to the answer is shown Figure 2.

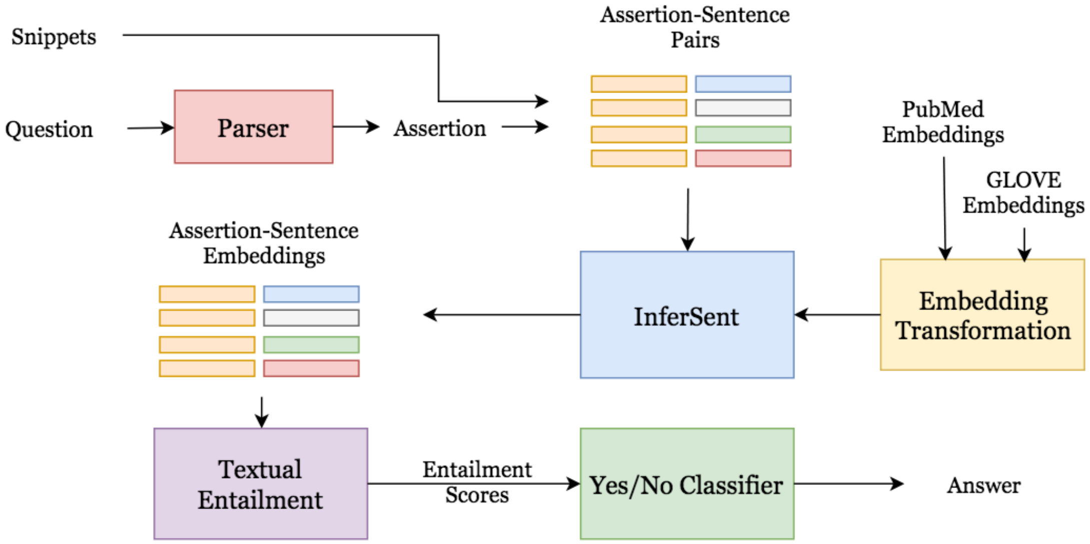

Figure 2: The complete system for yes/no answer classification using a question and relevant snippets

4.1.5 Experimental Details

For parsing the questions, we used BLLIP (Charniak-Johnson) reranking parser [5] and used the model `GENIA+PubMed` for biomedical text. For training the textual entailment classifier using *InferSent*'s sentence embeddings, we used Stanford's SNLI & Multi-NLI dataset [2] to achieve a test-set accuracy of 84.7%.

4.1.6 Results

The performance of the system on yes/no questions on the training set of phase 5b has been tabulated in table 2. While the accuracies are better than a random classifier, the task is far from being solved. Nonetheless, the classifier does handle the class bias in the training data and performance similarly on both the categories of answers, which also yields a high F1 score. This is an artifact of the classifier being unsupervised which does not bias it towards skewed distributions in the dataset.

This classifier achieved the best test performance with overall F1 of 65% on most recent phase (at the time of submission) i.e. phase 4 of BioASQ 6b, and second best test accuracy of 65.6% on phase 5 of BioASQ 5b (Table 2). While we implemented a simple heuristic based answer-classifier, we believe that a supervised classifier using the sentence embeddings as well as fine-tuning of the textual entailment classifier on BioASQ dataset would considerably enhance the overall performance of the system.

4.1.7 Error Analysis

A key challenge in improving the performance of the yes/no system is that it has a lot of modules in sequence, and hence, the attribution of errors in classification becomes difficult. However, we qualitatively analyze the kinds of classification errors depicted in Table 3.

It can be seen that for a lot of cases for which the model incorrectly incorrectly predicts the answer as *no*, the questions are usually for long. This is most likely because the entailment model

	Accuracy				F1
	Train		Test		Test
	5b	6b	5b Phase 5	6b Phase 4	6b Phase 4
Yes	0.57 (252/444)	0.58 (306/524)	-	-	0.7
No	0.59 (33/56)	0.63 (58/92)	-	-	0.6
Overall	0.57 (285/500)	0.59 (364/616)	0.65	0.67	0.65

Table 2: Class-wise accuracies on yes/no questions for training and test set of BioASQ Phase 5b and 6b

fails to perform well on long sentences, since it is trainined on SNLI which mostly consists of short sentences. However, the other types of errors i.e. when the model incorrectly predicts the answer as yes do no have any clear pattern.

	Question	Prediction	Answer
Correct	Does metformin interfere thyroxine absorption?	No	No
	Is the protein Papilin secreted?	Yes	Yes
Incorrect	Is the monoclonal antibody Trastuzumab (Herceptin) of potential use in the treatment of prostate cancer?	No	Yes
	Proteomic analyses need prior knowledge of the organism complete genome. Is the complete genome of the bacteria of the genus Arthrobacter available?	No	Yes
	Is amiodarone a class I anti-arrhythmic drug?	Yes	No

Table 3: Predicted and true answers by the yes/no classifier for some questions in BioASQ dataset 5b

4.2 Factoid & List Type Questions

Most of the state-of-the-art models for this task involve training end-to-end deep neural architectures to identify a subset of entities (or phrases) from the relevant snippets that are most likely to answer the question. But, owing to the small size of the dataset, we cannot effectively train such models on the BioASQ dataset. Hence, we adopted a two-stage approach that first finds a set of entities that could potentially answer the question and a supervised classifier to rank the entities on the basis of their likelihood of answering the question.

For devising the model and evaluation, we primarily focused on factoid type questions since the methodology for the list-type question would be largely similar and different only in the number of top entities returned.

4.2.1 Initial Hypotheses

Initially, we had the following hypotheses:

Hypothesis 1: A method that works for factoid type will probably work similarly with list type with a small group of changes (list size, etc).

Hypothesis 2: Extracting NER entities as our possible responses will be a sufficient approach in terms of generating a candidate space.

NER Tags	% of questions Exactly Answered	 Partially Answered	% of tokens extracted
PubTator	32.05	72.15	52.27
Gram CNN	34.90	99.03	94.97
LingPipe	26.67	76.75	11.06
Union	49.04	99.65	99.25
Intersection	16.29	38.00	3.33

Table 4: Baseline recall of different NER Taggers measured by the fraction of questions that can be answered by an ideal classifier if the candidates are chosen using the tagger. We also measure precision as the fraction of total unique tokens from the documents that are tagged.

Hypothesis 3: Ranking the sentences and extracting entities from the top-ranked sentences will give better results.

4.2.2 Candidate Selection

We found that the most critical step in the answer generation process is to identify the set of potential answer candidates that can be fed into a classifier or ranker to identify the best candidates. At first, in order to accomplish this, we used Named Entity Recognition (NER) taggers to form a set of candidate answers. The taggers that we used include Gram-CNN [29], LingPipe[3] and PubTator [25]. To analyze the effectiveness of these taggers, we performed an analysis on BioASQ training set 5b by evaluating the fraction of questions whose answers are included in the candidate entity set by the taggers.

Table 4 shows the relative performances of the three taggers, their union as well as intersection on train dataset of BioASQ 5b factoid type questions. A question is exactly answered if a tagger tags an entity that matches an answer exactly, and it is partially answered if there is a non-zero overlap with an entity tagged and an answer for the question. We can notice that PubTator and LingPipe have a good recall with relatively low precision, while Gram CNN has high recall but low precision. However, the final results with the Named Entity Taggers were not aligned with our expectations. This is mostly because the answers for BioASQ are usually a combination of BioNERs and complementary words, making it hard to define a pruning method that is able to yield satisfactory results. Surprisingly, a group of candidates formed of the 100 most frequent n-grams (n from 1 to 4) from the snippets' sentences were a better candidate group than the NER approach for our supervised ranking method and we decided to use the NER taggers as features instead of candidate groups.

4.2.3 Classification Features

Upon computing the set of candidate answers, we use the question q, set of relevant snippet sentences $\mathcal{S}$ and entity type t_i to devise a feature vector for each individual entity e_i that comprises the following features:

- BM25 Score: The BM25 scores for all the sentences are computed with the question as the query. Then, the scores of the sentence that contain the entity are aggregated to compute

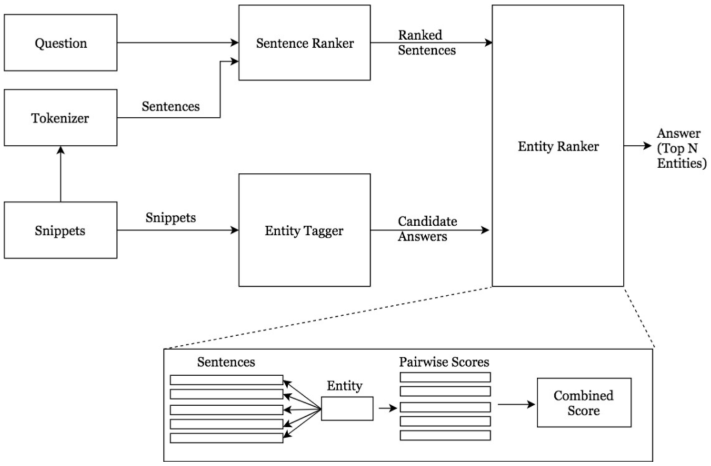

Figure 3: Unsupervised generation of factoid/list type answers using NER taggers and BM25 retrieval model

the BM25 score for the entity, i.e.

$$\text{Score}_{BM25}(e_i) = \sum_{s \in \mathcal{S}} \text{Score}_{BM25}(e_i) \cdot \mathbb{1}(s, e_i)$$

where $\mathbb{1}(s, e_i)$ is 1 iff sentence s has entity e_i.

- Indri Score: Computed in the same manner as BM25 score in (i)
- Number of Sentences: Number of sentences $s \in \mathcal{S}$ that contain the entity e_i
- NER Tagger: A multinomial feature that represents which tagger among PubTator, LingPipe and GramCNN the entity was extracted with. This feature is included to identify the relative strengths of the different taggers.
- Tf Idf: The aggregate Tf-Idf scores of the entity with $\mathcal{S}$ as the set of documents
- Entity Type: Is a boolean feature that is 1 if the type of the entity (for example, *gene*) is present in the question, and 0 otherwise.
- Relative Frequency: The amount of times the entity appears on the snippets' sentences divided by the total appearance of all of the relevant entities.
- Query Presence: Is a boolean feature that is 1 if the query contains the entity completely and 0 otherwise.

4.2.4 Unsupervised Ranking

As a baseline, we first present an unsupervised ranking system for the candidate answers. In this system, the snippet sentences are first ranked using the BM25 model. Then, for each entity, a score is computed by aggregating the BM25 scores of the sentences in which the entity is present. The rationale for this is that the entities in the top ranked sentences are more likely to be the answers. This entity score (which is equivalent to the BM25 score described in 4.2.3) is then used to rank the entities and return the top k entities as answers to the question. The overall unsupervised system is shown in Figure 3, with results tabulated in Table 7.

4.2.5 Learning To Rank

In order to rank the candidate entities in a supervised way, we used a ranking classifier based on the features described in 4.2.3. For ranking, we chose point-wise ranking classifiers over pair-wise and list-wise, because it yields similar results to ranking methods with a less time-consuming and computationally expensive approach. We are using a traditional SVM-Light [8] implementation for point-wise ranking. The data for supervision was derived from the actual answers and candidate entities were ranked based on their overlap with the actual answers.

Once we rank the entities, we use a naive approach of merely taking top 5 entities as answers for factoid type and top 10 for list-type. One could, however, devise a separate model for identifying the number of top entities to return as answers for the list-type answers.

We found that using just the NER entities as the answer candidates, the classifier could achieve an MRR of 0.06 on factoid type questions and an F-measure of 0.18 for list type questions. However, by having all the n-grams ($n = 1, 2, 3, 4$) from the snippets as candidate answers and using NER tags as LeToR features, the performance was improved to an MRR of 0.195 on Factoid type questions and an F1 score of 0.234 on List type questions. The results are summarized in Table 13 for BioASQ and in Table 5 for MS MARCO.

4.2.6 Error Analysis And Future Work

We can divide the type of mistakes that our model makes into three different parts, as seen on Table 6. The first one is when the model actually achieves the correct result as one of the returned answers but the option is not exactly reflected on the answer key (Type 1). The second type of mistake is when the model fails to collect a valid response, but one of the returned answers is very similar in structure or in semantic properties (Type 2). The third type of mistake is when our model completely fails to return relevant answers, not even getting close to a reasonable result.

Based on that, we can propose a few future directions. The first type of mistake is more related to the BioASQ evaluation structure than to an actual problem on our end, the solution, therefore, should be a proper channel of communication with the organization so we could explore a wider range of alternative responses for exact answers. For the second type of error, there are a few approaches that could significantly improve the model. The first one is to identify likely semantic clusters for a particular good answer and create more alternatives based on that. For instance, a wrong answer consisting of 'loss memory' could become a group of combinations: ('memory of loss', 'memory loss' and 'loss of memory'). Another alternative would be to use our abstractive summarization model to combine the important information from the top 5 answers and create new related responses based on that. For the third type of mistake, one interesting alternative would be to use some heuristics in order to reduce the number of candidates for each question. For example, a query about 'which gene' should only have 'genes' as candidates. The motivation is that with less possible candidates, the model would be able to rank them properly.

Metric	**Performance (%)**
Exact Precision	10.99
Soft Precision	21.98
MRR	15.20

Table 5: Performance of the entity-type answers on MS MARCO dataset. Here, exact precision refers to the the answer being the the top ranked candidate while soft precision refers to the answer being among one of the top 5 candidates

	Type 1	**Type 2**	**Type 3**
Question	*Which is the most prevalent form of arrhythmia worldwide?*	*what memory problems are reported in the " gulf war syndrome"?*	*what is the inheritance pattern of lifraumeni syndrome?*
Gold Answers	[u'atrial fibrilation', u'af']	[u'memory loss']	[u'autosomal dominant']
Our answer(s)	atrial fibrillation	loss memory	['family', 'predisposition', 'members family', 'dominant', 'gene']

Table 6: Error Analysis for factoid/list

Entities	Soft Accuracy (%)	MRR (%)
Pubtator	**7.14**	3.35
Lingpipe	4.58	2.70
Gram CNN	0.98	0.24
Union	1.05	0.38
Intersection	4.91	**3.68**

Table 7: Performance of unsupervised ranking model to identify the entities among the candidates from different taggers, for factoid type questions in BioASQ 5b dataset

4.2.7 Hypotheses Confirmation/Revision

After our results and error analysis, we can revisit our hypothesis and see what were the key learning points of this part of our project:

Hypothesis 1 (*confirmed*): The results for list type questions were very similar in comparison to factoid type.

Hypothesis 2 (*revised*): Although the evaluation document of BioASQ pointed for biological entities as the set of possible answers, most of the answers do not necessarily fit this category and there is a high variance in format/style for the ones that actually fit.

Hypothesis 3 (*revised*): Top-ranked sentences did not necessarily yield better results, but the ranking of the collected entities themselves delivered very strong results.

4.3 Summary Type Questions

This section describes our efforts to address the ideal answer category on BioASQ along with the description type category of MS MARCO. We proceeded with this answer type category with the specific hypothesis in mind. They are:

- Hypothesis 1: A QA system that provides accurate, non-repetitive answers must be supported by a strong Information Retrieval system.
- Hypothesis 2: Human readability is a harder problem to tackle with. Abstractive methods perform better on this front than the extractive methods

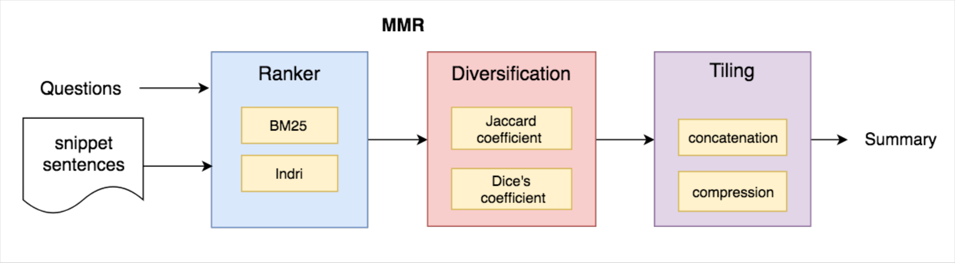

Figure 4: Pipeline for ideal answer generation

4.3.1 Extractive summarization

Our pipeline for ideal answers has three stages. The first stage involves pre-processing of answer snippets and ranking of answer sentences by various retrieval models described in the following sections. The retrieval model scores form the soft positional component introduced in the MMR algorithm. We perform sentence selection next, where we select the top 10 best sentences for generating an ideal answer. The third and final stage involves tiling together the selected sentences to generate a coherent, non redundant, ideal answer for the given question. The subsequent subsections explain the full pipeline for ideal answer type questions in detail (see Figure 4).

4.3.2 Question-Sentence Retrieval

In this section we describe various approaches which were adapted to improve the initial retrieval of candidate sentences. We used the standard BM25 algorithm with custom pre-processing to exclude medical entities from stop word removal.

BM25

BM25 [19] is a standard tf-idf based retrieval algorithm relying on bag of words approach for document retrieval. We considered every question to be independent and built an inverted index over the relevant snippets following the standard methods. Since the snippets are short paragraphs and the question is of moderate length, we tuned BM25 parameters accordingly. We customized the pre-processing by creating our own set of stop words that excluded certain bio-medical entities which might have been considered an English stop-word.

$$Score(D,Q) = \sum_{1}^{n} IDF(q_i)\frac{f(q_i,D)(k_1+1)}{f(q_i,D)+k_1(1-b+b.\frac{|D|}{avgdl})}$$

Indri

Indri [23] is more modern retrieval model based on the use of statistical language models and query likelihood. We assumed a uniform prior over the sentences and ranked the candidate sentences based on the probability of the question given the sentence. We employed a two-stage smoothing that considers characteristics of both the query (the question in this context) and answer sentences.

The Indri score for a candidate sentence is estimated in a collection (C) of snippets as follows:

$$p(q_i|d) = (1-\lambda)p_{mle}(q_i|d) + \lambda p_{mle}(q_i|C) \tag{1}$$

$$p_{mle}(q_i|d) = \frac{tf + \mu p_{mle}(q_i|C)}{length(d) + C} \tag{2}$$

$$p_{mle}(q_i|C) = \frac{ctf}{length(C)} \tag{3}$$

where, λ is the coefficient for linear interpolation based smoothing that accounts for question length smoothing. Since the questions are of moderate length, after tuning, the best value of λ is attained at 0.75

In equation 2, μ is parameter for Bayesian smoothing using Dirichlet priors used for sentence length normalization. Since sentences of snippets can be of varying lengths, after tuning, the best value of μ is attained at 5000.

Both of the above smoothing techniques do two different things, the mixture model (interpolation) compensates for differences in the word importance (gives idf-effects) and the Dirichlet prior improves the estimates of the sentence sample which supports our decision to use two stage smoothing.

4.3.3 Sentence Selection (MMR)

Once the top most relevant snippets have been chosen, we want to choose sentences from these snippets which are most relevant to a specific question. In this section we demonstrate how this selection is done.

We use the Maximum Marginal Relevance (MMR) algorithm [7] as the baseline for sentence selection. In contrast to the basic Jaccard similarity metric used in previous work [4], we experimented with other similarity measures which consistently perform better than the Jaccard baseline. MMR ensures the selected set contains non-redundant yet complete information. The sentences are selected based on two aspects, the sentence's relevance to the question and how different it is to the already selected sentences. At each step we select a document to append to the ranking based on the equation below.

$$d_i = \arg\max_{d_j \in R \setminus S}(\lambda \cdot sim(q, d_i) - (1-\lambda) \cdot max_{d \in \mathcal{S}}(sim_{sent}(d_i, d_j))) \tag{4}$$

We define a custom similarity metric between documents which uses positional values of sentences from the initial ranking as follows:

$$sim_{sent}(di, dj) = (1-\beta) \cdot (1 - \frac{rank(s_i)}{n}) + \beta \cdot sim(d_i, d_j) \tag{5}$$

Here, $sim_{sent}(di, dj)$ is the sentence to sentence similarity, $sim(q, di)$ is the question - sentence similarity, $rank(s_i)$ is the rank of the snippet which contains the sentence d_i, S are Sentences already selected for summary i.e. which are ranked above this position. In the above equation, we tried various metrics to account for the sentence to sentence similarity. In cases where β is non-zero, equation 4 is identified as our SoftMMR which includes soft scoring based on sentence position.

4.3.4 Dice's similarity Coefficient (DSC)

Dice's similarity Coefficient (DSC) [24] is a quotient of similarity between two samples and ranges between 0 and 1 calculated as It is used to compare similarity of two strings using bigrams. It

β	Configuration	Rouge-2	Rouge-SU4
-	baseline	0.7064	0.6962
0.5	BM25, Jaccard	0.7175	0.7110
0.5	BM25, Dice	0.7193	0.7106
0.6	BM25, Dice	0.7133	0.7053
0.6	BM25, Jaccard	0.7133	0.7053
0.5	**Indri, Jaccard**	**0.7206**	**0.7135**
0.5	Indri, Dice	0.7113	0.7052

Table 8: ROUGE scores for different experiments on similarity metrics for extractive summarization

Configuration	**Bleu 1**	**Bleu 2**	**Bleu 3**	**Rouge-L**
Wordlimit = 25, BM25, DuoSimilarity	0.215	0.139	0.109	0.175
Wordlimit = 50, BM25, DuoSimilarity	0.206	0.108	0.093	0.162

Table 9: Results on the MS Marco dataset

is different from the Jaccard coefficient which counts intersecting words only once in both the numerator and denominator.

$$dsc = \frac{(2 * n_t)}{(n_x + n_y)}$$

where n_t is the number of character bigrams found in both strings, n_x is number of bigrams in string x and n_y is the number of bigrams in string y.

4.3.5 Evaluation

BioASQ

The pipeline described above is primarily designed to improve the ROUGE evaluation metric [9]. Although a higher ROUGE score does not necessarily reflect improved human readability, MMR can improve readability by reducing redundancy in generated answers. Results for ideal answers for Task 5 phase b for BioASQ dataset are shown in Table 8. We also compare our results with other state of the art approaches in Table 13. Based on the results we accept our hypotheses 1 that a QA system needs to have strong IR.

MS MARCO

Now coming to the MS MARCO description dataset, the same pipeline couldn't be applied as is. Summary of BioASQ and Description of MS Marco are not exactly the same. Summary of BioASQ can be seen a subset of Description. The main difference being the average length of the description type answers. As mentioned in the dataset statistics, considering the number of sentences and average sentence length, we changed the number of words in summary from 200 to 20-25. That significantly improved our BLEU. Also, please note that we have performed the experiments on a smaller sample of the data. The results for description type questions are seen in the table 9.

4.3.6 Abstractive Summarization

Neural sequence-to-sequence models have provided a viable new approach for abstractive text summarization insteas of simply selecting and rearranging passages/sentences from the original text. However, these models have a problem where the summary generated looks like sentences stitched together and not readable as if generated by a human.To address this, we tried Pointer Generator Networks as mentioned in [20]

However, we could not get the expected results due to multiple reasons. We used a pre-trained model which was trained on CNN daily mail data which lead to many UNKs (Unknown words) due to difference in the Vocabulary. Hence, we couldn't reject or accept hypothesis 2 from our experiments due to insufficient metrics from the experiments. In future, we plan to add the coverage mechanism for penalizing repetitive sentences and train the network on a larger dataset like PubMed and probably have a better evidence for accepting our hypothesis 2.

4.3.7 Error Analysis

- Case 1: When the question asks to describe a very specific aspect/ affect.
  E.g.: *What causes genetic alterations in normal cells ?* Here the question asks to narrow a specific reason that explains a scenario. The answer generated by the system spoke about genetic alterations in normal cells and its characteristics but not causes.

- Case 2: When the answer is beyond the understanding of the question and content at a surface level.
  E.g.: *Why is albumin normally absent in urine?* Here, the system can't understand that it has to understand the word 'absent' and not match the sentences which has either only albumin or urine in it in a different context.

- Case 3: Inference type of questions.
  E.g.: *A went to New York and bought a house. Where is the house?* System couldn't infer that the house is in NY.

5 End-to-end System

By an *end-to-end* system, we refer to a question answering system where the individual parts of the system are not fine-tuned or modeled for specific sub-tasks, and a single model is trained to take the question as well as passages/documents as inputs and produce the answer as an output, without any explicit indication of the question/answer type. To train such a system, we make enhancements to the existing state-of-the-art end-to-end QA systems, which we describe in the subsequent sections.

5.1 Dynamic Co-Attention Networks

For span prediction, we implement a modified variant of the Dynamic Co-Attention Network (DCN) architecture [28]. This architecture is composed of an encoder segment and a decoder segment. The encoder segment first encodes the question and document word vectors with a bidirectional GRU. It then computes an attention both from the question text to the passage and from the passage to the question text to identify the most relevant information for each word in the passage-question pair. The computed attention over the words are fused to add contextual information to each word vector in the passage. In particular, the fusion process finds the soft dot attention from the question to the passage, from the passage to the question, and from the passage to the fused question-passage vectors:

$$\begin{aligned}
D, Q &: \text{The passage and question matrices, where each row is a word vector} \\
L &: \text{Dot affinity matrix} = DQ^T \\
C^Q &: \text{Fused question-passage attention matrix} = (L)Q \\
C^D &: \text{Fused passage-question attention matrix} = (L^T)D \\
C^{D\prime} &: \text{Fused passage-(question-passage) co-attention matrix} = (L)C^Q
\end{aligned}$$

This process is repeated with C^D replacing D and C^Q replacing Q to produce a new set of matrices C^{C^D}, C^{C^Q} and $C^{C^D}\prime$. The concatenation $[D, C^D, C^{D\prime}, C^{C^D}, C^{C^D}\prime]$ is finally passed to a second bidirectional GRU to produce a semantically rich and meaningful representations for each of word in the passage.

The decoder uses a GRU to iterate over the encoded passage vectors, predicting and iteratively improving its estimate of the start and end markers of the answer span:

$$\begin{aligned}
u_t &: \text{The encoding of the t}^{\text{th}} \text{ word in the passage} \\
h_t &: \text{The GRU hidden state at iteration t} = \mathit{GRU}(h_{t-1}, s_t, e_t) \\
\text{HMN} &: \text{Highway maxout network} \\
s_t &: \text{The probability vector for the location of the start marker at iteration t} = \text{HMN}(h_{t-1}, s_{t-1}, e_{t-1}) \\
s'_t & \text{ A one-hot vector corresponding to the location of } s_t \text{ with highest probability.} \\
e_t &: \text{The probability vector for the location of the start marker at iteration t} = \text{HMN}(h_{t-1}, s'_t, e_{t-1})
\end{aligned}$$

This iterative procedure enables the model to recover from initial local maxima corresponding to incorrect answers. This alleviates one of the major issues with question answering systems where they fail to make iterative refinements to the answer based on the context they learn. The traditional systems also fail to learn transitive relations for example they fail to identify the subjects which certain pronouns correspond to due to their single pass answer generation nature. On the other hand, the dynamic pointing encoder can resolve such relationships as it scans through the encoded passage vectors to predict the start and end markers.

5.1.1 Network Architecture

The encoder architecture is presented in Figure 5 and the Decoder architecture for DCN is presented in Figure 6.

5.1.2 Experiments

We first compute the GLoVe and CoVe word vectors for each word in the training and development data of the Stanford Question Answering Dataset (SQuAD) dataset, replacing all out of dictionary words with the zero vector. Then the network is trained with the Adam optimizer and the cross entropy loss for 40 epochs on a K80 GPU. A three-layer GRU network is used in the decoder

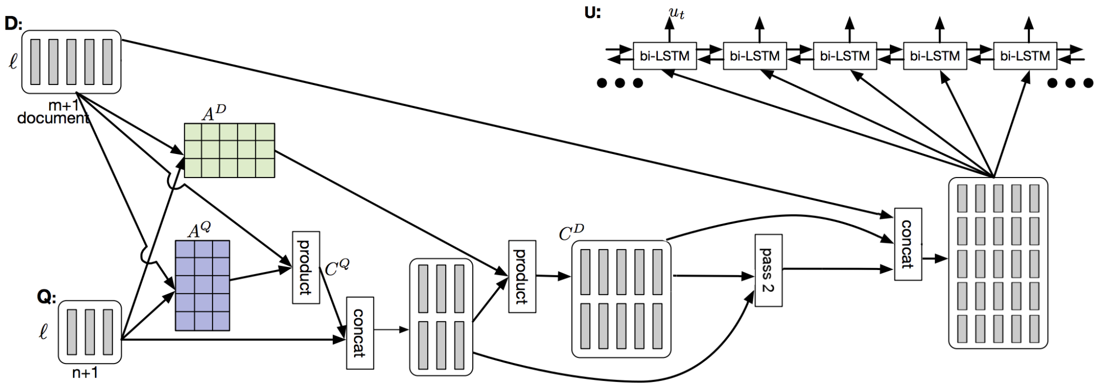

Figure 5: The DCN Co-attention Encoder [28]

Model Configuration	**Exact Match**	**F1 Score**	**Epochs**
DCN Vanilla	57.4	70.6	20
DCN Baseline	66.2	75.9	200 (w/ hyperparameter tuning)
DCN (Highway Networks)	59.8	71.8	20
DCN (CoVe)	62.1	74.4	20
DCN (ResNet encoder-decoder)	62.0	74.2	20
DCN (best grid search)	62.4	74.5	20
DCN (CoVe + Highway + deep resnet)	64.8 (qualifies for SQuAD leaderboard)	75.1	20

Table 10: Results on the SQuAD dataset

network for all experiments, but we adjust the number of hidden units for the GRU networks that exist in the architecture. Experiments with 30, 100 and 200 hidden units per GRU are performed. An annealed learning rate initialized at 0.002 and a weight decay of 0.0001 is used during the training process. Gradient clipping is employed to prevent large gradients from destabilizing the training process, and early stopping is employed to stop training once overfitting is detected.

5.2 Results on SQuAD

The results on SQuAD with all the different variety of approaches we tried are presented in Table 11. We present the results in a step by step fashion, showing how each of our modifications improves the results. We also note that we train our model for just 20 epochs compared to over 200 epochs used by the original model. Our best performing model qualifies to be ranked on the official SQuAD leader board.

An informal test is also performed with a small number of test cases not found in the data set. We find that the trained network has the capacity to answer these questions accurately as well. One out-of-dataset example is shown below:

This example demonstrates the ability of the network to resolve indirect relationships in the text, such as the chain connecting the phrase "The theory [is named after]" to its referent "The de BroglieBohm theory" to its alternative name "the pilot wave theory".

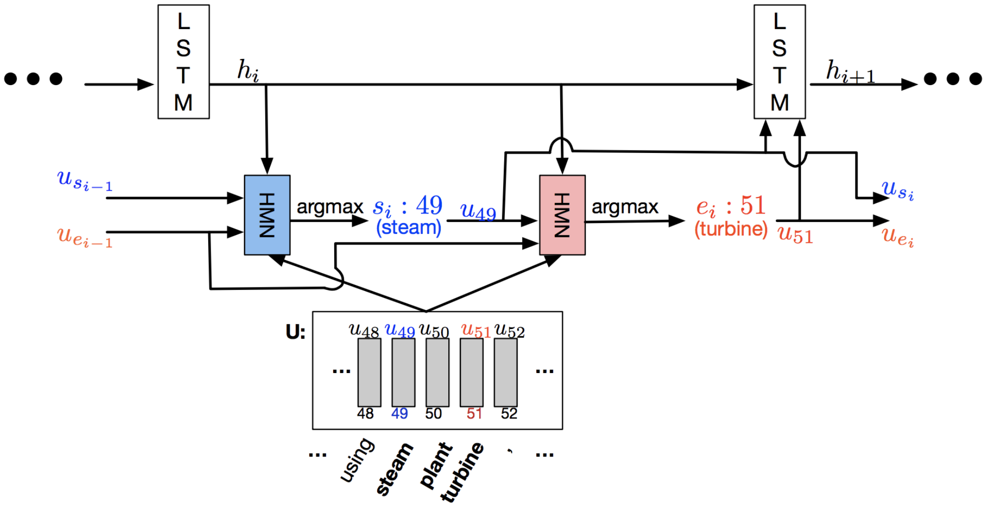

Figure 6: The Dynamic Decoder Network [28]

Model	Rouge	Bleu	Set Type
DCN Marco Net	34.7	31.0	Dev
DCN Marco Net	31.30	23.86	Test (Leaderboard)

Table 11: Results on the MS Marco dataset

5.3 Free Form Answer Generation

Free Form answer generation is a relatively unexplored problem with little existing literature. Trying to solve this problem directly is a fairly hard task and would require an extremely complex network architecture which can not just look at huge text sources to detect the most relevant parts, but also compose a human like answer. This is too hard for most networks to achieve. To reduce the complexity of the problem being dealt by a single network, we use the spans predicted by our Dynamic Co-Attention Network as input to the Free Form answer generator network. This makes the network more powerful by giving it more information about the content of the expected answer and it now has to use this content information to compose an answer.

Natural Language generation is known to be one of the hardest problems on the NLP community due to the inability of modern day architectures to model long term context and generate syntactic and semantically coherent sentences. One of the primary issues we notice with such architecture is

Document	The de BroglieBohm theory, also known as the pilot wave theory, Bohmian mechanics, Bohm's interpretation, and the causal interpretation, is an interpretation of quantum theory. In addition to a wavefunction on the space of all possible configurations, it also postulates an actual [...] The theory is named after Louis de Broglie (18921987) and David Bohm (19171992).
Question	After whom is pilot wave theory named after?
Predicted answer	Louis de Broglie

the ineffectiveness of the loss functions typically used in the seq2seq prediction networks typically used for this task. The seq2seq networks are usually based on LSTM encoder - decoder style architectures and typically penalize the generator using a cross entropy loss between the predicted and expected output word. Such work level loss metrics fail to capture global sentence level dynamics and completely miss out on enforcing syntactic and semantic coherence. We propose a novel loss metric here which we feel would be greatly useful in dealing with such issues.

First we introduce our generator network architecture. The architecture we propose is shown in Figure 7.

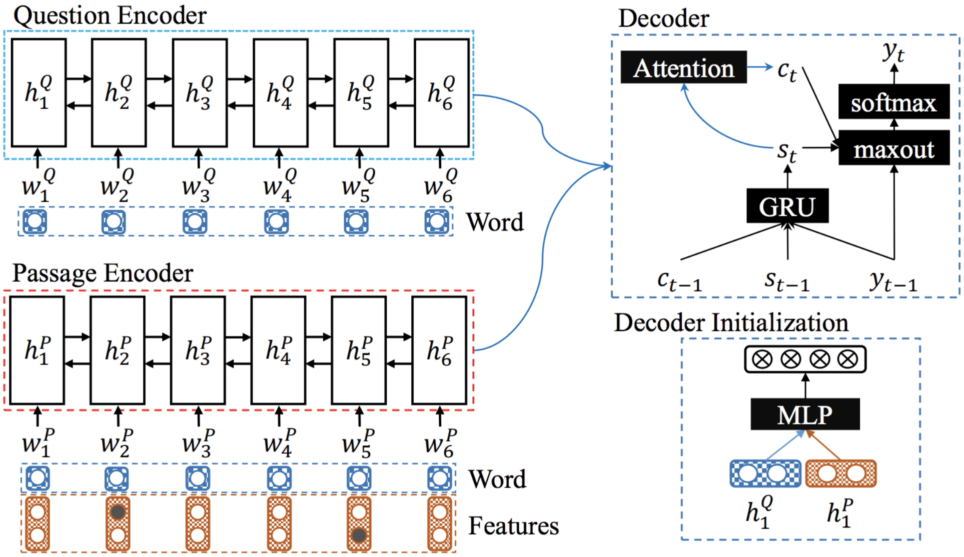
Question Encoder
h_1^Q h_2^Q h_3^Q h_4^Q h_5^Q h_6^Q
w_1^Q w_2^Q w_3^Q w_4^Q w_5^Q w_6^Q
Word
Passage Encoder
h_1^P h_2^P h_3^P h_4^P h_5^P h_6^P
w_1^P w_2^P w_3^P w_4^P w_5^P w_6^P
Word
Features
Decoder
Attention
c_t
y_t
softmax
maxout
s_t
GRU
c_{t-1} s_{t-1} y_{t-1}
Decoder Initialization
MLP
h_1^Q h_1^P

Figure 7: The Answer generator Network

The network consists of a question encoder network, a paragraph encoder network and an attention based decoder network.

The question and passage encoder are Bidirectional GRU's which take as input the question and passage word embeddings, one word at a time and generate the respective question and passage embedding. This is given by the equations:

$$h_t{}^P = BiGRU(h_{t-1}{}^P, [e_t{}^P, f_t{}^e, f_t{}^e])$$

$$h_t{}^Q = BiGRU(h_{t-1}{}^Q, e_t{}^Q)$$

here $h_t{}^P$ and $h_t{}^Q$ denote the passage and question encoder hidden state at time t, $e_t{}^P$ and $e_t{}^Q$ denote the word embedding of passage and question word at time t, $f_t{}^e$, $f_t{}^s$ are binary bits which are 0 or 1 depending of if the t^{th} word in the passage is the start or end of a span i.e. $f_t{}^e = 1$ if the t^{th} word is the end of a span and 0 otherwise. Similarly $f_t{}^s = 1$ if the t^{th} word is the start of the answer span and 0 otherwise.

The decoder is also a GRU which has a non linear combination of the question and passage encoding as its initial state denoted by:

$$d_0 = tanh(W_d[h_1{}^P, h_1{}^Q] + b)$$

The rest of the decoder can be represented as:

$$d_t = GRU(w_{t-1}, {c^P}_{t-1}, {c^Q}_{t-1}, d_{t-1})$$

where w_{t1} is the word generated at the previous time step, d_{t1} is the previous decoder state and ${c^P}_{t-1}$ is the passage context vector which is learnt by computing the attention of the decoder state over the passage states. Similarly ${c^Q}_{t-1}$ is the context vector computed by taking the attention of the decoder state over the states of the question encoder.

We experiment with two different types of attention mechanisms, which we call as the 'additive attention mechanism' and the 'dot attention mechanism'.
The additive attention mechanism is defined by the following equations.:

$$\begin{gathered} {s^P}_{t,j} = {{v^P}_a}^T tanh({W^P}_a(d_{t-1}) + {U_a}^P {h^P}_j) \\ {a^P}_{t,i} = softmax({s^P}_{t,j}) \\ {c^P}_t = \sum_{i=1}^{n} {a^P}_{t,i} \cdot {h^P}_i \end{gathered}$$

Similarly for question encodings:

$$\begin{gathered} {s^Q}_{t,j} = {{v^Q}_a}^T tanh({W^Q}_a(d_{t-1}) + {U_a}^Q {h^P}_j) \\ {a^Q}_{t,i} = softmax({s^Q}_{t,j}) \\ {c^Q}_t = \sum_{i=1}^{n} {a^Q}_{t,i} \cdot {h^Q}_i \end{gathered}$$

The dot attention on the other hand is defined by the equations:

$$\begin{gathered} {s^P}_{t,j} = (d_{t-1})^T {h^P}_j \\ {a^P}_{t,i} = softmax({s^P}_{t,j}) \\ {c^P}_t = \sum_{i=1}^{n} {a^P}_{t,i} \cdot {h^P}_i \end{gathered}$$

Similarly for question encodings:

$$\begin{gathered} {s^P}_{t,j} = (d_{t-1})^T {h^Q}_j \\ {a^P}_{t,i} = softmax({s^P}_{t,j}) \\ {c^P}_t = \sum_{i=1}^{n} {a^P}_{t,i} \cdot {h^P}_i \end{gathered}$$

These attention weighted question and passage encodings are fed to the decoder network.
To predict the next word we use the present decoder state, the question and passage context and the previous word and pass their combination through a Maxout layer, taking maxout over 2 successive units and then take a softmax over our vocabulary. This is represented by the equations:

$$\begin{gathered} r_t = W_r w_{t-1} + {U^P}_r {c^P}_t + {U^Q}_r {c^Q}_t + V_r d_t \\ m_t = [max r_{t,2j-1}, r_{t,2j}]^T \\ p(y_t|y_1, ..., y_{t-1}) = softmax(W_o \cdot m_t) \end{gathered}$$

Model	Factoid(MRR)	List(F1)
DCN MarcoNet	0.11	0.07

Table 12: Results on the Bioasq dataset

5.4 Results on MS MARCO

We use our free form answer generation network on the MSMarco V1 dataset. The results for the same are listed in Table **??**. We note that we are listed as one of the highest ranked single model on the actual MSMarco leaderboard.

5.5 Experiments and Results on BioASQ dataset

We further extend our model to evaluate it on the BioASQ dataset. We use our DCN model to perform the QA task on the BioASQ dataset. Presently, we only work with factoid and list type of questions. The architecture is exactly the same as the DCN encoder. In place of the decoder we simply use the attention values over the passage to predict our answers. For both factoid and list type questions, the passage entities which attain attention more than a threshold are chosen as the correct answers. For the Factoid type answers, as the ranking of these entities also matters we return them in the order of decreasing attention weights. The results of this model evaluated on the BioASQ dataset after pre-training on the SQuAD dataset are presented in Table 12. Note that presently the model uses word2vec and Cove embeddings and hence several of the Biological entities are actually not a part of the vocabulary which may partially explain the poor performance of the DCN model on the BioASQ dataset.

Model	Exact Answers Yes/No type Accuracy (%)	Exact Answers Factoid type MRR	Exact Answers List type F1 score	Ideal Answers All types ROUGE-2
[4]	-	-	-	0.653
[14]	**0.714**	0.272	0.187	-
[27]	-	**0.392**	**0.361**	-
Sarrouti and Alaoui usmba	0.461	0.207	0.243	0.577
BioAMA(Ours)	0.653	0.195	0.234	**0.721**

Table 13: Comparison of our question-type based QA model with other state of the art approaches on BioASQ 5b dataset

6 Discussion

In this section, we shall analyze our gross findings from building the two QA systems and experimenting on both the datasets. Having built a broad set of classifiers and modules, we also comment on our experiences in building these systems while highlighting our key learning from these experiences.

6.1 Model Comparison

The question-type based QA system that we built performs remarkably well on the BioASQ. This can be noted from the Table 13, where we compare our system with the state of the art systems on BioASQ. We can see that our system performs the best on Ideal Answers and is very close to the best performances on Exact Answers. The trends are very similar for BioASQ 6b as well, indicating that a question-type based system is very effective for BioASQ. However, our end-to-end system fails to perform well on the BioASQ, as shown in Table 12, which leads us to the conclusion that for smaller datasets, a question-type based modular system is typically a much better choice of QA system.

On the other hand, we note that the end-to-end systems that we proposed has much better performances on the large datasets such as MS MARCO and SQuAD at the ranks of some of the state-of-the-art systems, while our modular question-type based system fails to yield to similar performance. This should not be surprising, since our end-to-end QA system is heavily parametrized and a lot more capable, and having a large dataset would help us efficiently train the model. In the cases of smaller datasets, however, we would have to aid the model with inductive biases, and constrain it with very few parameters to strike a trade off between efficient training and optimal performance.

With these findings, we reject our general hypothesis that a holistic QA model can be trained to yield consistently high performances across multiple datasets, and conclude that a well performing QA system would have to be carefully chosen with the size and domain of the dataset as key considerations.

6.2 Lessons Learned

Building the two QA systems on two separate datasets involved a lot of design and implementation choices some of which, in hindsight, were sub-optimal. However, these choices coupled with our experimental findings have led us to have many insightful realizations, some of which are:

- Simple approaches can often yield more reliable and strong results than complicated ones. We realized that this is applicable to different question types:
  - In case of yes/no questions, a simple linear transformation for embedding worked much better than a neural network-based non-linear transformation for augmenting word embeddings
  - In the case of factoid/list questions, having n-gram based candidates worked much better than having NER-based candidates
- A long and in-depth dataset exploration is key to building features, especially for small datasets. However, it is important to not rely on specific aspects of a subset of the data as it is easy to overfit if the features come from a biased subset of the data. We learned this lesson when analyzing the data factoid-list type questions led us to have better features and candidates
- Additional computation time should be weighed against marginal performance gain. We learned this when we realized that some NER extraction techniques were far more expensive to compute than the marginal value they added to the ranking performance. This also applied to the choice of LeToR ranking algorithms where it was not feasible to efficiently fine-tune pairwise algorithms like SVMRank with a large number of answer candidates.

- Models for small datasets can be extremely sensitive of feature weighting and selection.
- Explicit Diversification methods need to be adapted to remove redundancy in the summary generated. This is applicable to the description category questions for any given type of dataset be it a large dataset like MS MARCO or a smaller dataset like BioASQ. The boost in ROUGE score and BLEU score we got from adapting Maximum Marginal Relevance is a strong indicator of the same.
- Even for datasets of large sizes, the QA system would depend heavily on the specific domain of the dataset. For example, PoS based features perform better for question type classification in Web search compared to BioMedical domain.

7 Individual Contributions

By virtue of having a tightly integrated system that was built collaboratively, we have had to work very closely with each other on almost all the modules involved. Though we all weighed in for the design and analysis of the systems, each one of us also assumed primary responsibility for implementation of specific parts of the system, which have been tabulated in Table 14.

Team Member	Yes/No Type	Factoid/List Type	Ideal Answers	End-to-end System
Nitish	✓	✓		
Gabriel		✓	✓	
Pranavi		✓	✓	
Vasu	✓			✓

Table 14: Key contributions from individual members of the team

8 Conclusion and Future Work

In this work, we propose two novel QA systems and present a comprehensive analysis of relative performances the systems two standard QA datasets (BioASQ and MS MARCO) with a primary aim to assess the feasibility of having a unified well-performing model across multiple datasets. Our key finding is that a modularized QA system involving question-type specific models performs quite well on smaller datasets (such as BioASQ) in comparison to the larger datasets. We also find that, while such systems can have similar performances on larger datasets as well, end-to-end systems can perform much better on these since they can leverage the large sizes of the datasets to a much greater extent.

To achieve state-of-the-art performances on BioASQ, we present an integrated framework for tackling both ideal and exact answer type questions and obtain state of the art results on the ideal answer type questions on the BioASQ dataset. In our framework for exact answers, we incorporate neural entailment models along with a novel embedding transformation technique for answering yes/no questions, and employ LeToR ranking models to answer factoid/list based questions. For ideal answers, we aimed at improving the Information Retrieval component of the extractive summarization. Although this improved ROUGE scores considerably, the human readability aspect of the generated summary answer was not improved to a great extent. We also build an end-to-end system with enhancements to DCN model as well as a free-form answer generation model, which achieves performances close to state-of-the-art systems for SQuAD and MS MARCO datasets.

As future directions, we believe that abstractive summarization based approaches like Pointer Generator Networks [20] and Reinforcement Learning based abstractive summarization techniques [13] can greatly improve the human readability of ideal answers for BioASQ dataset. However, to accomplish this, it is elemental to first identify more suitable evaluation metrics that quantify human readability of the ideal answers. For the yes/no answers, there is a lot of scope in using the entailment scores for for improving the classification scores. One such way is to build a supervised classifier that also accounts for the class bias in the dataset. It would also be helpful to heuristically create evaluation metrics for each module in the yes/no pipeline so as to individually improve the performance of each sub-module. Lastly, for the factoid/list/entity type answers, we believe that there is a great potential in exploring more candidate spaces and other LeToR ranking algorithms that also use NER-based features.

Despite the increasing progress in area of developing well-performing Question Answering systems, there is still a large scope of improvement for the existing models. We believe that our work helps better understand the nuances the existing QA systems and serves as a valuable step towards the larger quest for building intelligent QA systems for real-life use cases.

References

[1] Balikas, G., Krithara, A., Partalas, I., and Paliouras, G. Bioasq: A challenge on large-scale biomedical semantic indexing and question answering. In *Revised Selected Papers from the First International Workshop on Multimodal Retrieval in the Medical Domain - Volume 9059* (New York, NY, USA, 2015), Springer-Verlag New York, Inc., pp. 26–39.

[2] Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2015), Association for Computational Linguistics.

[3] Carpenter, B. Lingpipe for 99.99% recall of gene mentions. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop* (2007), vol. 23, pp. 307–309.

[4] Chandu, K., Naik, A., Chandrasekar, A., Yang, Z., Gupta, N., and Nyberg, E. Tackling biomedical text summarization: Oaqa at bioasq 5b. In *BioNLP 2017* (2017), Association for Computational Linguistics, pp. 58–66.

[5] Charniak, E., and Johnson, M. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA* (2005), pp. 173–180.

[6] Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. Supervised learning of universal sentence representations from natural language inference data. *CoRR abs/1705.02364* (2017).

[7] Forst, J. F., Tombros, A., and Roelleke, T. Less is more: Maximal marginal relevance as a summarisation feature. In *Advances in Information Retrieval Theory* (Berlin, Heidelberg, 2009), L. Azzopardi, G. Kazai, S. Robertson, S. Rüger, M. Shokouhi, D. Song, and E. Yilmaz, Eds., Springer Berlin Heidelberg, pp. 350–353.

[8] Joachims, T. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning* (1998), Springer, pp. 137–142.

[9] Lin, C.-Y. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out* (2004), p. 10.

[10] Liu, R., Hu, J., Wei, W., Yang, Z., and Nyberg, E. Structural embedding of syntactic trees for machine comprehension. *CoRR abs/1703.00572* (2017).

[11] Natural Language Computing Group, M. R. A. R-net: Machine reading comprehension with self matching networks. *ACL 2017* (2017).

[12] Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. Ms marco: A human generated machine reading comprehension dataset. *CoRR abs/1611.09268* (2016).

[13] Paulus, R., Xiong, C., and Socher, R. A deep reinforced model for abstractive summarization. *CoRR abs/1705.04304* (2017).

[14] Peng, S., You, R., Xie, Z., Wang, B., Zhang, Y., and Zhu, S. The fudan participation in the 2015 bioasq challenge: Large-scale biomedical semantic indexing and question answering. In *CLEF* (2015).

[15] Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)* (2014), pp. 1532–1543.

[16] Pyysalo, S., Ginter, F., Moen, H., Salakoski, T., and Ananiadou, S. Distributional semantics resources for biomedical text processing. In *Proceedings of LBM 2013* (2013), pp. 39–44.

[17] Radev, D., Hong Qi, Y., Wu, H., and Fan, W. Evaluating web-based question answering systems.

[18] Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. Squad: 100, 000+ questions for machine comprehension of text. *CoRR abs/1606.05250* (2016).

[19] Robertson, S., and Zaragoza, H. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr. 3*, 4 (Apr. 2009), 333–389.

[20] See, A., Liu, P. J., and Manning, C. D. Get to the point: Summarization with pointer-generator networks. *CoRR abs/1704.04368* (2017).

[21] Seo, M. J., Kembhavi, A., Farhadi, A., and Hajishirzi, H. Bidirectional attention flow for machine comprehension. *CoRR abs/1611.01603* (2016).

[22] Shen, Y., Huang, P., Gao, J., and Chen, W. Reasonet: Learning to stop reading in machine comprehension. *CoRR abs/1609.05284* (2016).

[23] Strohman, T., Metzler, D., Turtle, H., and Croft, W. B. Indri: a language-model based search engine for complex queries.

[24] Srensen, T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. In *Kongelige Danske Videnskabernes Selskab* (1948), pp. 1–34.

[25] Wei, C.-H., Kao, H.-Y., and Lu, Z. Pubtator: a web-based text mining tool for assisting biocuration. *Nucleic acids research 41*, W1 (2013), W518–W522.

[26] Weissenborn, D., Wiese, G., and Seiffe, L. Fastqa: A simple and efficient neural architecture for question answering. *CoRR abs/1703.04816* (2017).

[27] Wiese, G., Weissenborn, D., and Neves, M. L. Neural question answering at bioasq 5b. *CoRR abs/1706.08568* (2017).

[28] Xiong, C., Zhong, V., and Socher, R. Dynamic coattention networks for question answering. *CoRR abs/1611.01604* (2016).

[29] Zhu, Q., Li, X., Conesa, A., and Pereira, C. Gram-cnn: a deep learning approach with local context for named entity recognition in biomedical text. *Bioinformatics* (2017), btx815.