Pointing to On-Point Answers: An Approach for Query-Based Biomedical Summarization

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters

<table>
<thead>
<tr>
<th>Citable link</th>
<th><a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:38811563">http://nrs.harvard.edu/urn-3:HUL.InstRepos:38811563</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Terms of Use</td>
<td>This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA">http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA</a></td>
</tr>
</tbody>
</table>
Dedication

This thesis is dedicated to my parents Peter and Barbara, who have been a source of encouragement and inspiration to me throughout my life and have instilled in me a strong work ethic and sense of morality to reach for my dreams, and to my younger brother Trevor, who has always been creative and independent and whose do-it-yourself mentality I admire.
Acknowledgements

First, I would like to express my appreciation to my fantastic thesis advisor Professor Alexander Sasha Rush for challenging me to grow as a researcher and student this school year. I am grateful for the opportunity to join his lab and work directly with him during my senior year and appreciate his constructive feedback on the ways I can improve, from asking questions concisely while providing context, to communicating ideas effectively, to understanding models and concepts from papers more thoroughly, rather than rushing to invalid conclusions. I am also tremendously thankful for Professor Rush’s CS287 course on natural language processing (NLP) that I am taking this semester, which has exposed me to cutting-edge NLP research and mind-boggling problems; the combination of thesising and CS287 has been stretching my mind each day in the best ways. Furthermore, I appreciate Professor Rush’s organized weekly thesis meeting structure, in which I started off attempting to read landmark NLP papers and later learned to propose ideas in NLP and give research updates. I am thankful for Professor Rush’s patience, expert guidance, and strong support in empowering me to pursue my interests in both deep learning and biology.

I would also like to thank my incredibly supportive concentration advisor Professor David Cox, for giving me the opportunity to embark on computer science research for the first time in my entire life during my junior year, for inspiring me to dive deep into both biology and computer science, and for reading my thesis. Researching in the Cox Lab on connectomics has opened my eyes to the innovative and impactful work done at the intersection of computer vision and neuroscience that will one day elucidate mechanisms
behind neurological disorders. I appreciate the practical machine learning advice and tactics Professor Cox taught me as I worked on the connectomics project. Furthermore, I am grateful for the opportunity to have taken Professor Cox’s highly interactive and high-demand course MCB81: Fundamentals of Neuroscience, that has equipped me with neuroscience background to explore deeper questions about neurological diseases and psychiatric disorders. I was excited each day in class for MCB81 and also at Professor Cox’s popular office hours sessions where he discussed all sorts of futuristic and eye-opening ideas.

Also, I would like to extend my thanks to Professor Scott Kuindersma for reading my thesis and for being motivational and supportive, especially during my experiences in CS182 and later as a CS182 Teaching Fellow (TF). In CS182, I was initially scared of talking about technical material or discussing my final project ideas, but Professor Kuindersma’s enthusiastic nature and supportive attitude in answering my questions enabled me to become more comfortable sharing and workshopping ideas. As a CS182 TF under Professor Kuindersma’s leadership, I learned how to understand and channel what I had learned when I took CS182 into interpretable explanations in order to speak to students during weekly office hours. My experience as a Teaching Fellow was invaluable and special for me, and I am grateful to Professor Kuindersma for his trust in allowing me to TF.

Furthermore, I would like to thank Professor Barbara Grosz, who taught me to critically read 40 AI research papers in a single semester, ask relevant questions, and figure out how to challenge and extend papers. CS280r was a life-changing course that changed my thinking for artificial intelligence prob-
lems and beyond, including recursive p-beliefs and what collaboration and intention truly are. I especially thank Professor Grosz for being supportive of women interested in computer science and research, and for sharing her honest reflections and experiences from graduate school. It was an invigorating and rewarding experience working with Professor Grosz to develop my CS280r team’s final project on influencing agents for flocking into a real manuscript that was recently accepted for publication.

Additionally, I would like to thank Professor Regina Barzilay from MIT for encouraging me to pursue NLP research, for her life advice, and for introducing me to the exciting field of NLP. I was introduced to NLP through Professor Barzilay’s Advanced NLP course 6.864 at MIT and am grateful for Professor Barzilay’s high-energy and interactive teaching style from which I was exposed to a diversity of ideas in NLP. I remember how exciting it was to learn to build a RNN to identify genes in biomedical literature in one of the course problem sets. I am deeply inspired by Professor Barzilay’s research on using machine learning to advance cancer detection.

Moreover, I am incredibly grateful to graduate student Yoon Kim, who is also our teaching fellow for CS287, for his patience, support, and knowledge. Despite his immensely busy research schedule, Yoon has been supportive for both my project and learning in CS287 and has provided excellent guidance for my thesis. Yoon has helped me develop into a stronger thinker in a variety of ways, from filling in my knowledge gaps of deep learning methods to challenging me to creatively rethink how to frame research problems, and I am always excited to learn from the enlightening discussions with Yoon in Maxwell Dworkin.
I am thankful to Professor Michael Z. Lin and Dr. Jun Chu, my research mentors for four years from Stanford University, who stimulated my deep passion and curiosity for solving problems in biology. Under their mentorship, from sophomore year in high school to sophomore year spring/winter/summer breaks at Harvard, I explored the fascinating world of bioengineering and developed bright fluorescent proteins. I appreciate Professor Lin’s guidance and support for a young scientist each step of the way, from personally teaching me how to conduct PCRs and protein purification, to advising me on forming a strong scientific inquiry, to mentoring me in writing a manuscript and posters to present my results. I appreciate Dr. Jun Chu’s detailed daily instructions, creativity, and always-constructive feedback that has challenged me to become more organized and efficient with research. During these years, in addition to the many hours spent on the lab bench, I have learned to read biomedical papers and have become an active user (and now contributor) on PubMed, which has naturally driven my interest in biomedical question answering.

Furthermore, I would like to extend my thanks to my classmates Andrew Jin and Jacob Klegar for being amazing friends and problem set/project partners. It has been a rewarding experience investigating NLP problems with Andrew and Jacob, and I appreciate their strong support and belief in me.

Additionally, I would like to thank Raghu Dhara and Katherine Scott, who also wrote their theses with Professor Rush, for the insightful discussions and productive and fun thesis work sessions.

Moreover, I would like to thank my wonderful friends Evan Chen, Hugo
Yen, Charles Liu, Alex Xiao, Tomoka Kan, David Yang, and Noah Golowich for their help in proofreading my thesis, providing suggestions, or discussing ideas.

I would like to thank my family members for continuously inspiring me and unconditionally supporting in everything I wish to pursue. My family has always being supportive of me, especially while I wrote this thesis, from organizing delightful virtual study breaks to sing karaoke, to giving me invaluable life advice. Thank you for bringing so much happiness into my life.

Finally, I am grateful for my friends who have been there for me, supported me in countless ways in life, and grown up with me during my precious years in college.

This thesis would not have been possible without all of you, and I will forever treasure these valuable experiences. Thank you!
## Contents

1 Introduction .......................... 1
   1.1 Motivation .................. 1
   1.2 Contributions .............. 3

2 Background ..................... 5
   2.1 Query-based Summarization .... 5
      2.1.1 Formal Problem Definition .... 5
      2.1.2 Biomedical Setting .......... 5
   2.2 BioASQ Challenge ............. 6
      2.2.1 Domain Transfer and Adaptation .... 8
   2.3 Relevant Neural Architectures .... 9
      2.3.1 RNNs and LSTMs .......... 9
      2.3.2 Sequence-to-Sequence Network .... 11
      2.3.3 Attention ................. 14
      2.3.4 Pointer Network .......... 15

3 Approach .................... 17
   3.1 Biomedical Pointer Network (Biomed-Ptr) .... 17
      3.1.1 Sentence Ranking Problem .... 17
      3.1.2 Encoding with Continuous Bag of Words Representation 18
      3.1.3 Decoder with Modified Pointer Network .... 19

4 Experiments .................. 21
   4.1 Dataset ................. 21
   4.2 Data Pre-Processing ....... 21
1 Introduction

1.1 Motivation

Question answering (QA) is an exciting task in natural language processing (NLP). A wide array of QA datasets have been generated for diverse applications, such as crowd-sourced Wikipedia questions in the Stanford Question Answering Dataset (Rajpurkar, 2016), real-user search engine queries in the MS-MARCO Dataset (Nguyen et al., 2016), questions about CNN articles in NewsQA (Trischler, 2016), and reading comprehension questions on children’s books (Children’s Book Test). In this thesis, I explore question answering in the biomedical domain, which poses interesting deep learning challenges due to the small number of training samples and domain expertise required.

Biomedical QA is an interesting challenge relevant to both biomedical and computer science research. In the biomedical research process, significant time is spent finding and reading relevant research papers. In recent years, there has been a significant increase in the number of biomedical research papers being published online. The United States National Library of Medicine at the National Institutes of Health maintains a free search engine called PubMed that contains references and abstracts on life sciences and biomedical topics (NLM, 2018). PubMed has over 27.3 million records dating back to 1966, with 13.1 million records containing research abstracts. The fast-paced growth of information in PubMed makes it difficult for biologists to manually find relevant answers for questions on PubMed. A method that improves efficiency of the literature search process would undeniably
expedite the research process.

From the computer science perspective, biomedical QA is also an intriguing challenge: how can deep learning architectures be designed for domain-specific cases with limited amounts of training data? Currently, the most heterogeneous training set with the largest number of question-answer pairs for biomedical QA is the BioASQ dataset, which only contains 1,799 biomedical questions. This dataset contains biomedical questions, selections of abstract snippets from which an answer can be inferred, and an ideal answer. In comparison to existing large-scale question-answer datasets such as SQuAD and MS-MARCO, the size of the BioASQ dataset is significantly smaller, and the nature of the summary answers present in BioASQ are more complex. Unlike the short and continuous single-span answers in SQuAD, summary answers from BioASQ are multi-span and composed of multiple sentences from disjoint abstract snippets.

In the BioASQ dataset, including the summary questions, there are four types of questions (of increasing complexity): yes-no, list, factoid, and summary. Each training sample contains a question, an answer, and a relevant context passage from a PubMed abstract from which an answer can be inferred. For biomedical QA, end-to-end neural models have only been used to answer questions with simpler reasoning levels (list/factoid) whose answers are direct single-span sub-sentence extractions. Furthermore, the end-to-end models for list/factoid involve transfer learning from large-scale question answer models trained on the SQuAD dataset, such as FastQA. These models also do not provide multi-span answers, in which answers are constructed from different parts of the passage. Deep learning methods have yet to be
applied to answer the more complex summary-based questions (query-based summarization), which require context from throughout the passage and only have ideal answers. Since there is not necessarily an exact sentence match between the answer and the passage, ideal answers are difficult to generate. Therefore, in addition to the aforementioned challenges of domain expertise and a small number of training samples, query-based biomedical summarization produces unique challenges for machine learning models due to its multi-span and non-exact answers.

These challenges relevant to natural language processing and biomedical research are the driving forces for my thesis.

1.2 Contributions

First, this paper describes a new approach for query-based biomedical summarization using an end-to-end deep learning model called Biomedical Pointer Network (Biomed-Ptr). To our knowledge, deep learning has not been used for answering biomedical summary questions, and existing methods rely on a series of traditional statistical techniques that require high domain expertise and feature engineering.

Second, Biomed-Ptr extends on Pointer Network to directly factor in questions in attention and uses biomedical word2vec to point to sentences in the passage. Biomed-Ptr is based on CBOW encodings and does not rely on a RNN-based encoder from the traditional sequence-to-sequence model.

Lastly, many question answering models are not suitable for multi-span answers. Biomed-Ptr uses sentence ranking to extract answers consisting of multiple sentences, not necessarily contiguous (Many previous models only
predicted one small continuous span of words). This means that the sentences
extracted will already be syntactically correct and contain specific language
that can answer the summary well.
2 Background

2.1 Query-based Summarization

2.1.1 Formal Problem Definition

Query-based summarization involves constructing an answer to a question from a relevant context passage. In this paper, I focus on extractive query-based summarization, where the question’s answer is not generative; the answer is constructed directly from sentences in the context passage. Given a question $Q$ (represented as a vector), a relevant context passage $P = (p_1, \ldots, p_n)$ with $n$ sentences (each represented as a vector, hence $P$ is a matrix), the objective is to predict answer indices $a = (a_1, \ldots, a_k)$, where each $a_i$ for $i \in \{1, \ldots, k\}$ is an integer representing the index of the sentence in $P$. Our aim is to learn a probabilistic model of $a$ given $P$ and $Q$. Thus, the probability of generating the answer sequence can be modeled as:

$$p(a|P, Q) = \prod_k p(a_k|a_1, a_2, \ldots, a_{k-1}, P, Q).$$

The query-based summarization task is then equivalent to finding the sequence of answer indices $a_l$ for each of $N$ training pairs $(Q_l, P_l)$ (for $l = 1, \ldots, N$) that minimizes

$$-\sum_{l=1}^{N} \log p(a_l|P_l, Q_l).$$

2.1.2 Biomedical Setting

The BioASQ Challenge provides summary-type biomedical questions, context passages that consist of sentences extracted from relevant PubMed ab-
stracts, and ideal answers generated by domain experts in biology. In the challenge, while the ideal answers are not necessarily generated word by word from the snippets, as the domain experts are free to rephrase and shorten snippets for better readability or conciseness, it is given that all information required to compose the answer are present within the context passages. While the ideal answers are not exact matches, a simple analysis of the ideal answers and context passages revealed that many of the sentences can be directly extracted or extracted by approximate string matching of answer and passage sentences. Therefore, I frame this query-oriented summarization problem as extractive.

2.2 BioASQ Challenge

The BioASQ Challenge is a question answering and information extraction challenge (Tsatsaronis et al., 2015). Each training sample in the BioASQ dataset contains a question, a gold answer, and a collection of relevant context snippets from PubMed abstracts from which an answer can be inferred. The questions are categorized into different question types: yes-no, list, factoid, and summary.

- Yes-no questions are questions that require “yes” or “no” answers. A sample question is: “Are stress granules involved in the pathogenesis of Amyotrophic Lateral Sclerosis?” The exact answer is: “yes.”

- List questions require a list of entity names, numbers, or similar short expressions as an answer, such as a list of gene names. A sample list question is: “Which proteins participate in the formation of the
ryanodine receptor quaternary macromolecular complex?” The exact answer for this question is “Ryanodine receptor, Calsequestrin, Triadin, Junctin.”

- Factoid questions require a specific entity name, a number, or a similar short expression as an answer, such as the name of disease or drug. A sample factoid question is: “In which cells are A-type lamins expressed?” The exact answer is: “late differentiating primary cells.”

- Summary questions are questions that are answered via a short summary text that includes the most relevant information from the passage. A sample summary question is: “What is the aim of the Human Chromosome-centric Proteome Project (C-HPP)?” and its ideal answer is: “The chromosome-centric human proteome project aims to systematically map all human proteins, chromosome by chromosome, in a gene-centric manner through dedicated efforts from national and international teams.”

Yes-no, list, and factoid questions have exact and ideal answers, in which the exact answer can be directly matched word-for-word from the context passage. Summary questions, however, have only ideal answers, in which the ideal answer can be inferred from the contents of the context passage but may not necessarily have an exact match. Summary answers are multi-span; their sentences may not be consecutive in the context passages and can be from distinct PubMed abstracts.

In this work, I focus on predicting answers to summary questions. For evaluation of summaries, a model’s candidate summaries are compared to the
gold summaries using a statistical $n$-grams-based metric called ROUGE, and human judgment of information recall, information precision, information repetition, and readability.

2.2.1 Domain Transfer and Adaptation

Most deep learning models require large amounts of human-labeled training data, which limits their applications in settings such as in biomedical QA. Collecting data is laborious, time-consuming, expensive, and requires domain expertise. Even though some domains contain more training data, it is challenging to build models that are generalizable across different domains. Naively applying a model trained on source domain $A$ directly to target domain $B$ often leads to severe drops in model performance, which is why many efforts in domain-specific applications have centered around domain transfer. The objective of transfer learning is to transfer some of the model’s learned capabilities to another domain. Domain transfer can be used in order to train a model on source domain $A$ that contains numerous training examples, and subsequently adapt the model to make quality predictions for source domain $B$ with limited training data.

Recently, many open-domain QA models have achieved strong performance on key question answering datasets such as SQuAD, which has introduced the idea of applying domain transfer from the open-domain QA models to domain-specific QA challenges. In a previous BioASQ Challenge, Wiese et al. (2017) applied supervised domain transfer on a trained, state-of-the-art end-to-end neural QA model called FastQA (Weissenborn et al, 2017) for answering list and factoid questions. After training FastQA, Wiese
et al. (2017) conducted fine-tuning on trained FastQA to answer BioASQ list and factoid questions using biomedical word2vec embeddings and objective functions to discourage catastrophic forgetting. Catastrophic forgetting is a phenomenon in which the model forgets what it has learned from source domain $A$.

For my thesis, I did not explore domain transfer from an open domain end-to-end model similar to Wiese et al.’s (2017) work due to the more complex characteristics of summary questions compared to list and factoid questions in the BioASQ Challenge. Most of the strong-performing open domain end-to-end models can only predict a single-span short output, or are trained on a source domain that differs too drastically from BioASQ in nature.

### 2.3 Relevant Neural Architectures

Now, I will provide background on some of the neural architectures that are important to this work.

#### 2.3.1 RNNs and LSTMs

In traditional neural networks, the assumption is that all inputs are independent from one another. In contrast, Recurrent Neural Networks (RNNs) leverage sequential information to capture information of all the inputs seen over time.

RNNs are called *recurrent* because they perform the same calculations for each element of a sequence that helps to provide an output. Whereas in theory RNNs should be able to encode information across long contexts, in
practice, RNNs can only recall information from a few timesteps. A Long Short Term Memory Network (LSTM) is a variant of the recurrent neural network (RNN) that can better handle long-term dependencies; it has been shown to remember relevant information in the sequence for an extended number of timesteps. The LSTM’s dynamics thereby enable learning of long-term dependencies across more context compared to a traditional RNN.

LSTMs contain repeating modules (units) where each unit is composed of a cell and gates that compute activations to regulate the cell state: an input gate, an output gate, and a forget gate. The input gate controls the flow of new values into the cell, the forget gate controls whether values stay in the cell or are “forgotten,” and the output controls how the cell’s values will impact the output activation of the unit. The “long term memory” is stored in a vector of memory cells. Due to their unique cell state setup that retains information across timesteps, LSTMs do not face the problem of gradient vanishing through backpropagation through time.

Mathematically, a LSTM updates hidden state $h^t$ and long-term memory...
Figure 2: Long Short-Term Memory Cell Schematic (Retrieved from Graves et al., 2013).

The cell $c_t$ is updated as follows:

$$
\begin{bmatrix}
i \\ f \\ o \\ g
\end{bmatrix} =
\begin{bmatrix}
\text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \text{tanh}
\end{bmatrix}
\begin{bmatrix}
T_{2n,4n} \\
h_{t-1}^l \\
h_{t-1}^l
\end{bmatrix}
$$

$$
c_t^l = f \circ c_{t-1}^l + i \circ g
$$

$$
h_t^l = o \circ \tanh(c_t^l).
$$

### 2.3.2 Sequence-to-Sequence Network

Before the powerful paradigm of the sequence-to-sequence (Seq2Seq) model, existing architectures of recurrent neural networks (RNNs) posed limitations. RNNs are neural networks that sequentially processes one element of the input sequence at a time to compute the probabilities of the next element.
of the output sequence. The memory state of the RNN is initialized with a vector of zeros and is updated after reading each word. Therefore, a vanilla RNN’s versatility is limited by the length and order of sequence prediction, in which each input corresponds to an output at each timestep. Seq2Seq models free us from the single vanilla RNN’s restrictions of sequence length and order by including two RNNs: an encoder and a decoder, to allow for different sequence lengths and orders between the source and target sequences, which enables flexibility that has revolutionized performance on countless natural language processing problems. For example, Seq2Seq plays a huge role in improving performance in summarization and machine translation, in which summaries and outputs may have different sentence lengths and different word orderings than the original text.

In a Seq2Seq network, the first RNN, termed an encoder, encodes the source sequence into a context vector that represents the source sequence. For each input in the source sequence, the encoder outputs an output vector and a hidden state, and for the next timestep, the hidden state is fed back into the encoder to compute the following input’s output. The context vector of the source sequence is derived from the encoder’s last hidden state, which theoretically encodes context across the entire source sentence. In the next stage, the context vector is fed into the second RNN, known as the decoder RNN, which generates predictions for the next tokens in the target sequence. The decoder’s initial hidden state is initialized to be the context vector.

At each timestep $i$ of decoding, the decoder takes in the previous hidden state as well as an input token (initially a special start token), and the decoder terminates upon generating a special stop token. After the start token, there
Figure 3: (a) **Sequence-to-Sequence**: Encoder RNN (blue) feeds Decoder RNN (purple) to generate prediction from fixed dictionary. (b) **Ptr-Net**: Encoder RNN (blue) feeds Decoder RNN (purple) to produce an attention vector over all inputs, generating a prediction with dictionary size equal to input length. (Retrieved from Vinyals et al, 2015).
are two commonly used ways to derive the input token. First, the input token can be the outputted token predicted by the decoder at timestep $i-1$. Second, instead of using the predictions as input, another approach called “teacher forcing” involves feeding the decoder the real target tokens in order to “correct” the decoder and ensure that the decoder stays on track. Teacher forcing allows for faster convergence, but also creates instability as the model relies on the teacher initially and does not properly fully learn meaning to construct the target sequence on its own.

As demonstrated by Vinyals et al. (2015), to put this into formal terms, let us define $\mathcal{P} = (p_1, \ldots, p_n)$ to be a sequence of $n$ input vectors and $\mathcal{C}^\mathcal{P} = (C_1, \ldots, C_{m(\mathcal{P})})$ to be a sequence of $m(\mathcal{P})$ indices between 1 and $n$. Given a sample of training data ($\mathcal{P}, \mathcal{C}^\mathcal{P}$), a Seq2Seq model with parameters $\theta$ estimates and learns $\theta^*$ to maximize the log of the conditional probability

$$p(\mathcal{C}^\mathcal{P}|\mathcal{P}; \theta) = \prod_{i=1}^{m(\mathcal{P})} p(C_i|C_1, \ldots, C_{i-1}, \mathcal{P}; \theta).$$

In this Seq2Seq model, the output dictionary size for $C_i$ is fixed and equal to $n$, which requires a separate model to be trained for each $n$. This is problematic in scenarios where problems have output dictionaries that depend on input sequence length.

### 2.3.3 Attention

In addition to the widely-utilized sequence-to-sequence architecture, another approach that has revolutionized the performance of deep learning models in natural language processing is the attention mechanism. Attention overcomes the challenge of compressing the source sequence into a fixed length context.
vector and burdening the fixed-size context vector to encode the entire input sequence. At a high-level, attention enables a model to place emphasis and shine light on all elements seen thus far that are critical for predicting the next element. With attention, it is feasible to retrieve encoded information from earlier timesteps to provide more detailed and relevant context for the decoder to reference. One form of attention is called multilayer perceptron (MLP) attention. Let us define the encoder and decoder hidden states as \((e_1, \ldots, e_n)\) and \((d_1, \ldots, d_m)\), respectively. For MLP attention, the attention vector \(a^i_j\) is computed by the decoder at timestep \(i\) as follows:

\[
\begin{align*}
  u^i_j &= v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in \{1, \ldots, n\} \\
  a^i_j &= \text{softmax}(u^i_j) \quad j \in \{1, \ldots, n\} \\
  d'_i &= \sum_{j=1}^{n} a^i_j e^j.
\end{align*}
\]

Here the softmax function normalizes vector \(u^i\) (of length \(n\)) to act as the attention weights over the \(n\) encoder hidden states and thus highlights the importance of each previously seen encoder input in the sequence. We let \(v\), \(W_1\), and \(W_2\) denote learnable parameters of the attentional model, while \(d'_i\) and \(d_i\) are concatenated to be the hidden states for the decoder’s next timestep.

### 2.3.4 Pointer Network

Pointer Network (Vinyals et al., 2015) is a neural network architecture that is useful for solving problems in which elements of an output sequences are discrete and derived only from elements of the input sequence. The model’s output dictionary size is the number of elements in the input sequence. From
a high-level perspective, the Pointer Network (Ptr-Net) essentially enables an answer sequence to be constructed by “pointing” at each timestep to the relevant tokens in the input sequence. The Pointer Network achieves “pointing” by modifying the traditional sequence-to-sequence architecture to creatively use the attention mechanism to select elements of the input sequence as output.

Ptr-Net is a simple modification of the attentional Seq2Seq model that overcomes the limitation of a vanilla Seq2Seq model’s fixed output vocabulary. Instead of using Seq2Seq’s softmax distribution over a fixed output vocabulary for the final prediction layer \( p(C_i|C_1, \ldots, C_{i-1}, \mathcal{P}) \), we use the attention mechanism to model prediction as the last step. Unlike the attentional Seq2Seq model, the attention weights serve as pointers to input elements, rather than weights to bias the decoder hidden state via blending.

\[
    u_j^i = v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in (1, \ldots, n)
\]

\[
    p(C_i|C_1, \ldots, C_{i-1}, \mathcal{P}) = \text{softmax}(u^i)
\]

By directly leveraging tokens in the input sequence, Ptr-Nets have dynamic vocabularies and can be used for situations in which there is a variable number of target classes in each step of the output.
3 Approach

Previously, I have defined the formal problem statement for query-based biomedical summarization and discussed the Pointer Network (Vinyals et al., 2015) structure which lays the foundation of the approach. I will now present the architecture for Biomedical Pointer Network.

3.1 Biomedical Pointer Network (Biomed-Ptr)

Biomed-Ptr modifies Pointer Network and extends its use for applications in query-based biomedical QA. I implemented a modified version of Pointer Network to factor in a question for attention and constructed continuous bag of words sentence representations using biomedical word2vec to conduct multi-span sentence-level extractions.

3.1.1 Sentence Ranking Problem

Many existing large-scale question answering models are not suitable for multi-span answers and only predict one small continuous span of words. This is problematic for biomedical summary QA because sentences in the summary question’s ideal answer may not be consecutive in the context passages and can be from distinct PubMed abstracts. Therefore, I frame the task of biomedical query-based summarization discussed in the Problem Statement section as a sentence ranking problem. Rather than matching a single span of words in the context passages, as conducted by many question-answering models that train on SQuAD (Rajpurkar et al., 2016) and MS-Marco (Nguyen et al., 2016), Biomed-Ptr may construct answer summaries.
by matching disjoint sentences for multi-span extraction.

Since each of the summary question’s ideal answers are at least one sentence long and can be approximately matched to sentences in the context passages, we can transform the summarization task into an extractive one. By framing this task as a sentence ranking problem, I overcome the challenges of predicting multi-span answers. Furthermore, sentence-level ranking means that in the final decoding step, the sentences extracted will already be syntactically correct and contain specific language that can answer the summary well. Therefore, the objective is to identify sentences in the context passage that can best answer the question.

There are two steps in this approach: encoding the question and passages, and answer extraction from the passages.

3.1.2 Encoding with Continuous Bag of Words Representation

Recent question answering models have used Pointer Network as a decoder to an RNN encoder in the conventional Seq2Seq framework, such as in S-Net (Tan et al., 2017) and Match-LSTM + Answer Pointer (Wang et al., 2016).

However in this approach, Biomed-Ptr uses a continuous bag of words (CBOW) representation for encoding sentences in the question, passage, and answers (used for teacher forcing). In CBOW for sentences, fixed-size sentence vectors are generated by summing all the fixed-sized word vectors from a sentence. The rationale behind using pre-trained CBOW embeddings instead of an encoder RNN is to reduce the number of training parameters and decrease model complexity, which is essential for the small size of the BioASQ dataset. The word vectors used to construct sentence-level CBOW
representations are pre-trained biomedical word2vec vectors. The biomedical word2vec embeddings are of dimension \( d = 200 \) and were previously trained on all available PubMed abstracts.

Let \( n \) be the number of sentences in each context passage, \( m \) be the number of sentences in the summary answer, and \( d \) be the biomedical word2vec sentence embedding size. Using CBOW, the encoded passage is represented by matrix \( \mathbf{P}_e \in \mathbb{R}^{(n+1) \times d} \); the passage is concatenated with an end of summary vector of size \( d \) to indicate termination during decoding. The embedded question is represented by vector \( \mathbf{Q}_e \in \mathbb{R}^d \), and the embedded answer is represented by a matrix \( \mathbf{A}_e \in \mathbb{R}^{m \times d} \). It is important to note that \( \mathbf{A}_e \), which is fed into the decoder network during teacher forcing, should not be confused with the target vector of answer indices \( \mathbf{a} \) used to compute training loss. As stated previously, we represent the target answer indices as a vector \( \mathbf{a} \in \mathbb{R}^A \), where \( A \) is number of sentences in the answer. Post-CBOW, the training set across \( N \) examples now includes quadruples in the form of

\[
\{(\mathbf{P}_e^l, \mathbf{Q}_e^l, \mathbf{A}_e^l, \mathbf{a}_l)\}_{l=1}^N.
\]

### 3.1.3 Decoder with Modified Pointer Network

In the next step, the CBOW embeddings are fed into the modified Pointer Network. In Biomed-Ptr, a simple change is made to modify Pointer Network to factor in the question at each timestep. By adding the question in Pointer Network’s attention calculations, Biomed-Ptr is less complex and does not require a separate encoder RNN for sentence-passage matching. Biomed-Ptr takes in as input: \( \mathbf{P}_e \), \( \mathbf{Q}_e \), and \( \mathbf{A}_e \).
1. **Long Short-Term Memory Network** The modified Pointer Network includes a Long Short-Term Memory Network (LSTM) followed with attention. In the modified Pointer Network, the LSTM takes in as input its previous hidden state, initialized to a zero vector, and an element of $A^e$ as the decoder input in sequential order to conduct teacher forcing. At timestep $k$, the hidden state $d_k$ is outputted by the LSTM from the $k$th sentence vector in $A^e$.

2. **Modified Attention** Next, in order to generate the $k$th answer token as indicated by $a_k$, the attention mechanism is used to calculate an attention weight vector $\beta_k \in \mathbb{R}^{n+1}$ based on the question, passage, and decoder input. Recall that there are now $n+1$ sentence vectors in the context passage, due to the addition of the end of summary vector. The new calculation to compute the attention weight vector $\beta^k_j$ is as follows:

$$u_{k,j} = v^T \tanh(W_1 P^e_j + W_2 d_k + W_3 Q^e)$$

$$\beta_k = \text{softmax}(u_k)$$

$$\beta_{k,j} = p(a_k = j | a_1, ..., a_{k-1}, P^e, Q^e)$$

At each timestep $i$ during decoding, the modified Pointer Network will point to one of the sentences in the context passage or the end of summary token. To train the model across $N$ examples, we minimize the loss function:

$$- \sum_{l=1}^{N} \log p(a_l | P_l, Q_l).$$
4 Experiments

4.1 Dataset

The training data was the BioASQ Challenge Task 5B training set (Tsatsaronis et al., 2015). This training set consists of 1,799 biomedical questions, where each question is associated with a question type, URL’s of relevant PubMed articles and relevant snippets from those articles.

4.2 Data Pre-Processing

The maximum number of sentences in the context passage is 146. Most questions contained answers within the first 30 sentences in the context passage, so I filtered out questions which had answers beyond 30 sentences and terminated the passage lengths at 30. After filtering out questions with missing components, the training data resulted in 1,708 samples. I tokenized the question, ideal answers, and passages into sentences, and then used approximate string matching based on edit distance to match passage sentences to ideal answers in order to derive an answer index for the extractive summarization approach.

4.3 Implementation Details and Experiment Settings

4.3.1 Model

I implemented Biomed-Ptr in PyTorch and trained models using a Standard NC6 (6 vcpus, 56 GB memory) on Microsoft Azure. The model was trained on all questions with ideal answers, and tested on summary questions.
The biomedical word embeddings are from Pavlopoulos et al. (2014) and 200-dimensional word2vec embeddings (Mikolov et al., 2013) that were trained on approximately 10 million PubMed abstracts. The model was trained with a batch size of 128 for a maximum of 200 epochs with early stopping, where the parameters of the model were saved if the model improved test perplexity after the epoch. For decoding, greedy search was used to select each answer index to produce text summaries.

I conducted hyperparameter tuning and experimentation with learning rate, hidden layer size, number of hidden layers, and dropout. The best performing model was trained with learning rate 0.01, hidden size of 100, 1 layer, and 0.75 dropout. The loss function used was Cross Entropy Loss with ignore index initialized to the padding token, and the optimizer used was Adam. For the LSTM in Biomed-Ptr, the weights were initialized uniformly at random between $[-0.04, 0.04]$. The gradient norms were clipped at 10.

4.3.2 Evaluation

In experiments, model performance is evaluated based on (1) perplexity and (2) ROUGE-2 and ROUGE-SU4 scores. Optimizing ROUGE directly often does not lead to good summaries as evaluated by human judgement (Paulus et al., 2017) and perplexity has been shown to be a useful metric for indirectly improving ROUGE scores and human judgement (Liu et al., 2018). Perplexity is a common metric for language modeling that can be calculated as the exponentiation of entropy, or the exponentiated negative average log-likelihood, and ROUGE scores are used for comparing candidate and reference summary answers. ROUGE-2 and ROUGE-SU4 are the spe-
specific evaluation metrics used in the official BioASQ Challenge. ROUGE-\(n\) score calculates the proportion of \(n\)-grams co-occurring in the candidate and reference summaries. ROUGE score for candidate summary \(S\) and reference summary \(R\) is calculated:

\[
\text{ROUGE-N} (S | \text{Refs}) = \frac{\sum_{R \in \text{Refs}} \sum_{g_n \in R} C(g_n, S, R)}{\sum_{R \in \text{Refs}} \sum_{g_n \in R} C(g_n, R)}.
\]

### 4.3.3 Baseline Methods

The baseline methods are methods that performed well in the BioASQ challenge: OAQA and BM25+UMLS. Both models do not involve end-to-end deep learning. In fact, most biomedical QA systems rely on traditional statistical approaches without deep learning, such as the BioASQ baseline structure proposed by Weissenborn et al. (2013). Unlike end-to-end deep learning, these models rely on many discrete steps such as named-entity recognition, question classification, and candidate answer scoring, and this process requires significant feature engineering and biomedical domain-speciﬁc experts. The OAQA (Yang et al., 2016) design, which achieved strong performance in the 5th round of BioASQ for summarization, uses a biomedical parser, entity tagger, and a thesaurus.

### 4.4 Results

The results show significant overﬁtting on the train dataset, where after several epochs, the validation perplexity ceased to decrease past 21 (Figure 5) as the training loss continued to drop each epoch towards 0 (Figure 4).

To improve the performance of Biomed-Ptr, I experimented with the following: (1) hidden size of LSTM, (2) number of LSTM layers, (3) dropout,
Figure 4: Training loss for Biomed-Ptr.

![Mean Loss Learning Curve](image1)

Figure 5: Perplexity over epochs for Biomed-Ptr.

![Perplexities](image2)
and (4) learning rate. I experimented with hidden size and the number of layers of the LSTM to decipher the best hidden size that would capture enough feature representation without overfitting to the small training set. I also experimented with dropout, a helpful technique for reducing overfitting in neural networks. Dropout works by zeroing out units in the LSTM at random during training. Finally, I experiment with learning rate.

While hyperparameter tuning indeed yielded in less overfitting compared to the first iteration of Biomed-Ptr (not shown), significant overfitting remained present post-tuning.

In Figures 6-8, there appears to be two clear trends. First, for all hyperpa-
parameter configurations, including ones with extreme regularization measures, the validation perplexities remain around 22 (Figures 6-8). Since the perplexities reported are the perplexities for the epoch that showed the last validation perplexity improvements before early stopping, Figures 6-8 do not show the training perplexity nearing 0 after more epochs. The second trend is that by increasing the hidden size dimensions of the LSTM, the number of epochs required to reach the best validation perplexity decreases.

Even though the results for the experimentation are similar, the parameters that yielded the best performance are hidden size of 100, 1 layer, and 0.75 dropout. While results for all configurations are similar, it makes sense that the simpler models with fewer parameters (ie: smaller hidden size and fewer

<table>
<thead>
<tr>
<th>Dropout</th>
<th>train ppl</th>
<th>val ppl</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>18.69</td>
<td>22.07</td>
</tr>
<tr>
<td>0.5</td>
<td>18.29</td>
<td>22.04</td>
</tr>
<tr>
<td><strong>0.75</strong></td>
<td><strong>17.11</strong></td>
<td><strong>21.87</strong></td>
</tr>
<tr>
<td>0.9</td>
<td>18.1</td>
<td>21.88</td>
</tr>
<tr>
<td>0.95</td>
<td>18.24</td>
<td>21.9</td>
</tr>
</tbody>
</table>

Figure 8: Testing dropout rates for LSTM.

<table>
<thead>
<tr>
<th>Model</th>
<th>ROUGE-2</th>
<th>ROUGE-SU4</th>
</tr>
</thead>
<tbody>
<tr>
<td>OAQA</td>
<td>0.689</td>
<td>0.6789</td>
</tr>
<tr>
<td>BM25 + UMLS</td>
<td>0.5773</td>
<td>0.5756</td>
</tr>
<tr>
<td>Biomed-Ptr</td>
<td>0.3420</td>
<td>0.3340</td>
</tr>
</tbody>
</table>

Figure 9: Comparison of ROUGE scores for BioASQ Challenge Task 5B
layers), as well as increased regularization, could potentially result in better performance, given the small size of the training dataset. Manual inspection of the prediction results showed that the model was indeed producing diverse predictions without early termination, even though some of the samples had repetition of sentences.

Furthermore, at the moment, Biomed-Ptr’s ROUGE scores are significantly below the baseline (Figure 9).
5 Discussion

5.1 Challenge of Small-Sized Domain-Specific Datasets

In this work, I have presented and explored a new approach for tackling query-based biomedical summarization using Biomed-Ptr. During training and evaluation, despite hyperparameter tuning and regularization, it was challenging to fine-tune Biomed-Ptr to reduce overfitting and generalize to new training examples. Biomed-Ptr’s low-achieving performance during the training and development of neural models in this thesis may have been due to BioASQ’s (1) small dataset size and (2) expert-level biomedical background required to answer questions. With only around 1,700 questions, coupled with the fact that the questions are domain-specific, it is not feasible to apply transfer learning to the biomedical domain via an established large-scale question-answering model, such as in Wiese et al.’s (2017) approach to transfer from FastQA from SQuAD. Transfer learning from existing QA datasets for BioASQ summary questions would be difficult as existing QA datasets do not have significant overlap with BioASQ with regards to BioASQ’s heavy jargon, multi-span answers, necessitated context, and formal style.

This thesis is based on the idea that in order to train a model for good performance on a small dataset, it may be helpful to use a less complex model with fewer parameters as a starting point to prevent overfitting. It may be useful to reduce the number of parameters as much as possible to avoid learning a monstrous and complex, non-interpretable network. For these reasons, I modified Pointer Network to be as simple as possible: to use CBOW instead of the traditional encoder LSTM in Seq2Seq and to simply weigh the
question in attention rather than using a Match-LSTM to find connections between the question and passage. Furthermore, instead of concatenating multiple embedding types such as character-level embeddings and normal word2vec embeddings to the biomedical word2vec embeddings, I use only pre-trained biomedical word2vec embeddings in order to simplify the model and decrease the number of parameters.

The experimental results, however, indicate severe challenges in tuning Biomed-Ptr to avoid overfitting, as well as the poor performance of Biomed-Ptr compared to traditional non-deep learning methods. These traditional methods break down the QA pipeline into multiple small steps and piece together distinct statistical modules that integrate domain-specific information to craft a final answer solution. For example, the Yang et al. (2015) system includes NLP annotators, machine learning models for search result scoring, and collective answer re-ranking to generate a final answer. The hypothesis followed by these models is highlighted in Ferruci et al. (2009) and Yang et al. (2015): in order to optimize for challenges such as BioASQ, the best approach is to carefully engineer an architecture that utilizes many combinations of cutting-edge learning components, rather than relying on a “single magic component” (Yang et al., 2015). An end-to-end neural model, such as Biomed-Ptr, would indeed be considered a “single magic component.”

The traditional non-deep learning approaches’ high performance may be due to their engineered domain-specific features crafted with guidance from domain experts, which are not available in deep learning. Despite Ferruci et al. and Yang et al.’s potential lack of confidence in the performance of deep networks for BioASQ, it is important to conceive creative strategies to include
more biology context in deep learning models without the careful collection of hard-coded engineered features. Therefore, the question boils down to: how can we create more biology context without manually engineering biology-specific features?

5.2 Generating Biology Context

As Biomed-Ptr’s current ROUGE scores are significantly below the two baselines, developing strong sentence representations can be a promising avenue to improve model performance. Improved sentence representations may help to encode increased biology context to facilitate better QA. In this thesis, a continuous bag of words sentence representation was utilized due to its simplicity, robustness, and ease of implementation. The CBOW-based Biomed-Ptr is an initial starting point for implementing Biomed-Ptr’s framework to explore different sentence representations. Moving forward from CBOW, a major challenge will be to traverse beyond simple supervised transfer learning via adding static or non-static biomedical word2vec embeddings.

When manually reading through the questions in the BioASQ dataset, it is evident that the dataset contains significant jargon from terminology about specific biological molecular events and genetic pathways, which makes the questions difficult to interpret for non-domain experts. If the biomedical text is esoteric to most humans, it is easy to imagine how challenging it must be for a model to decipher biological context via deep learning, which drives an urgent need for transferring more biology context in models.

One way to potentially add more biology context is to improve the sentence embeddings beyond biomedical word2vec. Improved sentence repre-
sentations to feed into Biomed-Ptr can be useful for multiple reasons. First, during pre-processing, when ideal answers are matched to exact sentences in the context passage via edit distances, the matches are not exact and may be prone to error. Thus, the model that we train on extracted exact answers to generate summaries does not reflect the contextual meaning of the true ideal answer. By using improved sentence representations, we can then match ideal answer sentences and passage sentences using the overlap between the representations, which will hopefully provide more contextual overlap than edit distances, and thus enable the model to optimize towards generating output more similar to the real ideal answer. Second, improved sentence representations will enable better matching between question and the passage sentence during the attention step.

Next, I would like to explore building variational autoencoders in order to create a strong representation of biomedical sentences. A variational autoencoder (VAE) is a generative network, where the encoding network generates latent vectors that is constrained to roughly follow a unit Gaussian distribution. To generate a sentence, the latent vector is sampled from the unit Gaussian and passed into the decoder network. The latent vector can be extracted as the new sentence representations. The VAE’s loss function is composed of (1) the generative loss, which is a mean squared error between the reconstructed sentence and actual sentence, and (2) KL divergence, which measures how closely the latent variables match a unit Gaussian. A VAE trained on PubMed abstracts offers an opportunity for improved sentence representations with improved biology context that may improve matching between (i) questions and answers, and (ii) ideal answers and exact answers.
6 Conclusion

In this thesis, I investigated the potential of using deep learning for query-based biomedical summarization. To our knowledge, past efforts have explored applying deep learning to answer biomedical questions with simpler reasoning levels, such as list and factoid questions, but not to biomedical summary QA, potentially due to challenges posed by the small quantity of training examples and required domain expertise. I developed Biomed-Ptr: a model that outputs multi-span query-based summaries of biomedical text by extending Pointer Network. Differences between Biomed-Ptr and existing QA models include its question-biased attention mechanism, use of a CBOW model instead of a LSTM encoder, use of biomedical word2vec embeddings, and sentence-level ranking framework. Biomed-Ptr thus shedded light on the possibility of modeling the summary-based biomedical answers using sentence ranking, which enables multi-span answering and reduces model complexity.

This thesis explored deep learning for a small, multi-span, summary-based, and domain-specific question answering dataset. Future research involves building improved sentence representations using variational autoencoders trained on PubMed abstracts, that will hopefully encode more biology context.
7 Works Cited


Pavlopoulos, I., Kosmopoulos, A., and Androutsopoulos, I., 2014. Continuous space word vectors obtained by applying word2vec to abstracts of biomedical articles http://bioasq.lip6.fr/info/BioASQword2vec/

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J., 2013. Distributed representations of words and phrases and their compositionality.
In Advances in neural information processing systems (pp. 3111-3119).


