tds Published in Towards Data Science · Following

Meraldo Antonio · Follow
Aug 28, 2019 · 5 min read · Listen

THE DEFINITIVE GUIDE TO BIDAF — PART 1 OF 4

An Illustrated Guide to Bi-Directional Attention Flow (BiDAF)

This article illustrates the workings of BiDAF, an NLP model that has pushed the envelope in the Question and Answer domain.

The year 2016 saw the publication of BiDAF by a team at University of Washington. BiDAF handily beat the best Q&A models at that time and for several weeks topped the leaderboard of the Stanford Question and Answering Dataset (SQuAD), arguably the most well-known Q&A dataset. Although BiDAF's performance has since been surpassed, the model remains influential in the Q&A domain. The technical innovation of BiDAF inspired the subsequent development of competing models such as ELMo and BERT, by which BiDAF was eventually dethroned.

When I first read the original BiDAF paper, I was rather overwhelmed by how seemingly complex it was.

> BiDAF exhibits a modular architecture — think of it as a composite structure made out of lego blocks with the blocks being "standard" NLP elements such as GloVe, CNN, LSTM and attention. The problem with understanding BiDAF is that there are just so many of these blocks to learn about and the ways they are combined can seem rather "hacky" at times. This complexity, coupled with the rather convoluted notations used in the original paper, serves as a barrier to understanding the model.

In this article series, I will deconstruct how BiDAF is assembled and describe each component of BiDAF in (hopefully) an easy-to-digest manner. Copious amount of pictures and diagrams will be provided to illustrate how these components fit together.

Here is the plan:

- Part 1 (this article) will provide an overview of BiDAF.

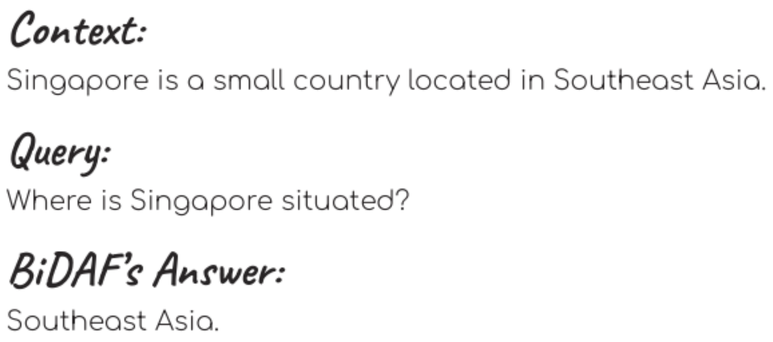
- Part 4 will talk about **the modeling and output layers.** It will also include a recap of the whole BiDAF architecture presented in a very easy language. If you aren't technically inclined, I recommend you to simply jump to part 4.

BiDAF vis-à-vis Other Q&A Models

Before delving deeper into BiDAF, let's first position it within the broader landscape of Q&A models. There are several ways with which a Q&A model can be logically classified. Here are some of them:

- **Open-domain** *vs* **closed-domain.** An *open-domain* model has access to a knowledge repository which it will tap on when answering an incoming *Query*. The famous IBM-Watson is one example. On the other hand, a *closed-form* model doesn't rely on pre-existing knowledge; rather, such a model requires a *Context* to answer a Query. A quick note on terminology here — a "Context" is an accompanying text that contains the information needed to answer the Query, while "Query" is just the formal technical word for question.
- **Abstractive** *vs* **extractive.** An *extractive* model answers a Query by returning the substring of the Context that is most relevant to the Query. In other words, the answer returned by the model can always be found verbatim within the Context. An *abstractive* model, on the other hand, goes a step further: it paraphrases this substring to a more human-readable form before returning it as the answer to the Query.
- **Ability to answer non-factoid queries.** *Factoid* Queries are questions whose answers are short factual statements. Most Queries that begin with "who", "where" and "when" are factoid because they expect concise facts as answers. *Non-factoid* Queries, simply put, are all questions that are not factoids. The non-factoid camp is very broad and includes questions that require logics and reasoning (e.g. most "why" and "how" questions) and those that involve mathematical calculations, ranking, sorting, etc.

So where does BiDAF fit in within these classification schemes? **BiDAF is a closed-domain, extractive Q&A model that can only answer factoid questions.** These characteristics imply that BiDAF requires a Context to answer a Query. The Answer that BiDAF returns is always a substring of the provided Context.



Context:

Singapore is a small country located in Southeast Asia.

Query:

Where is Singapore situated?

BiDAF's Answer:

Southeast Asia.

An example of **Context, Query** and **Answer.** Notice how the Answer can be found verbatim in the Context.

Another quick note: as you may have noticed, I have been capitalizing the words "Context", "Query" and "Answer". This is intentional. These terms have both technical and non-technical meaning and the capitalization is my way of indicating that I am using these words in their specialized technical capacities.

With this knowledge at hand, we're now ready to explore how BiDAF is structured. Let's dive in!

Overview of BiDAF Structure

> BiDAF's ability to pinpoint the location of the Answer within a Context stems from its layered design. Each of these layers can be thought of as a transformation engine that transforms the vector representation of words; each transformation is accompanied by the inclusion of additional information.

1. Embedding Layers

BiDAF has 3 embedding layers whose function is to *change the representation of words* in the Query and the Context from strings into vectors of numbers.

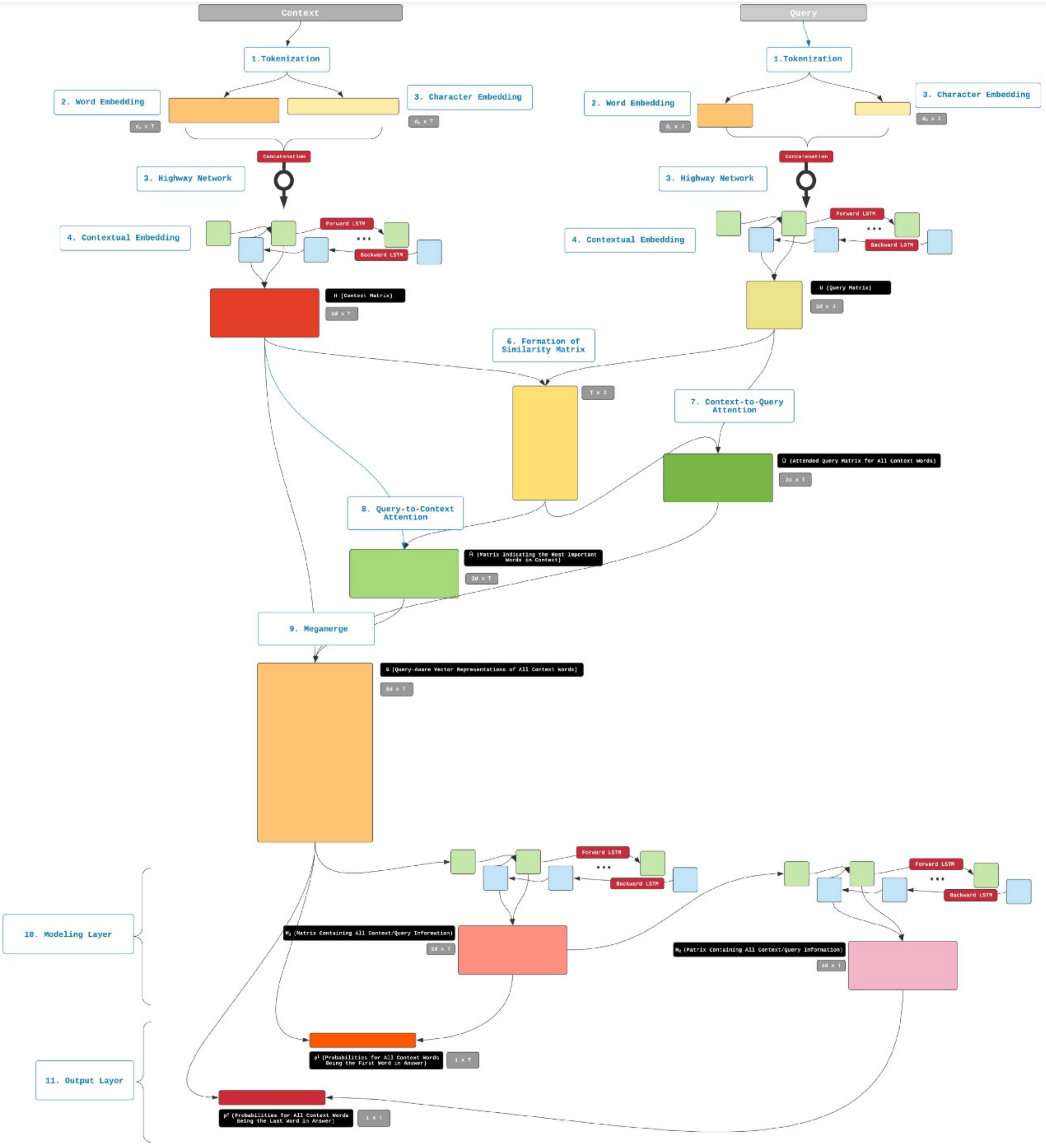
2. Attention and Modeling Layers

These Query and Context representations then enter the attention and modeling layers. These layers use several matrix operations to *fuse the information contained in the Query and the Context.* The output of these steps is another representation of the Context that contains information from the Query. This output is referred to in the paper as the "Query-aware Context representation."

3. Output Layer

The Query-aware Context representation is then passed into the output layer, which will *transform it to a bunch of probability values*. These probability values will be used to determine where the Answer starts and ends.

A simplified diagram that depicts the BiDAF architecture is provided below:


Context
1.Tokenization
2. Word Embedding
d_1 x T
3. Character Embedding
d_2 x T
Concatenation
3. Highway Network
4. Contextual Embedding
Forward LSTM
Backward LSTM
H (Context Matrix)
2d x T
Query
1.Tokenization
2. Word Embedding
d_1 x J
3. Character Embedding
d_2 x J
Concatenation
3. Highway Network
4. Contextual Embedding
Forward LSTM
Backward LSTM
U (Query Matrix)
2d x J
6. Formation of Similarity Matrix
T x J
7. Context-to-Query Attention
$\tilde{U}$ (Attended Query Matrix for All Context Words)
2d x T
8. Query-to-Context Attention
$\tilde{H}$ (Matrix Indicating the Most Important Words in Context)
2d x T
9. Megamerge
G (Query-Aware Vector Representations of All Context Words)
8d x T
10. Modeling Layer
Forward LSTM
Backward LSTM
M_1 (Matrix Containing All Context/Query Information)
2d x T
Forward LSTM
Backward LSTM
M_2 (Matrix Containing All Context/Query Information)
2d x T
11. Output Layer
p^1 (Probabilities for All Context Words Being the First Word in Answer)
1 x T
p^2 (Probabilities for All Context Words Being the Last Word in Answer)
1 x T

Architecture of BiDAF

If all these don't make sense yet, don't worry; in the next articles, I will delve into each BiDAF component in detail. See you in part 2!

References

[1] Bi-Directional Attention Flow for Machine Comprehension (Minjoon Seo *et. al*, 2017)

If you have any questions/comments about the article or would like to reach out to me, feel free to do so either through LinkedIn or via email at meraldo.antonio AT gmail DOT com.

Special thanks to @Raimi Karim who helped me review this article!