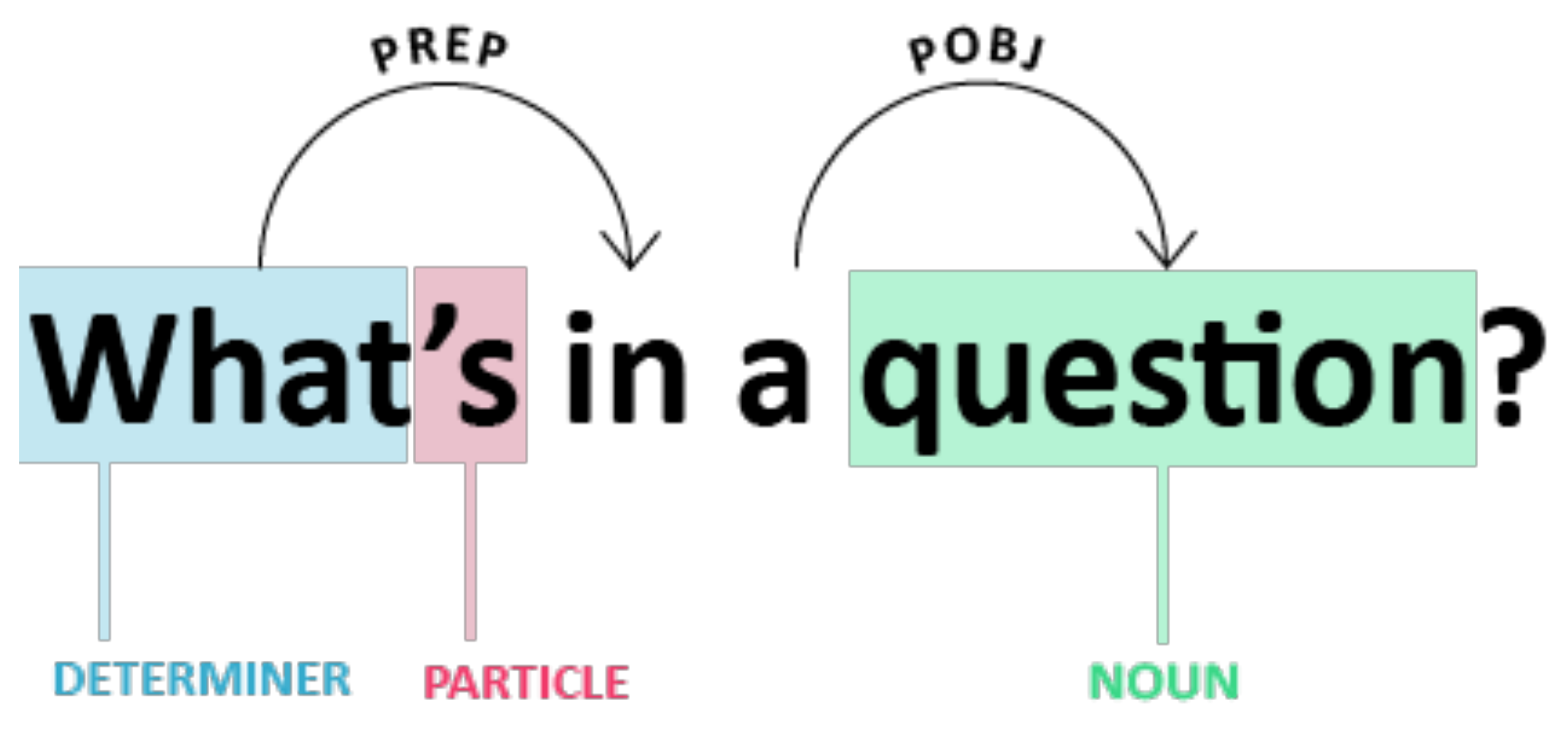
Tulane University
SCHOOL OF
SCIENCE AND ENGINEERING

What's in a question?



Dung Ngo Computer Science & Mathematics
Abe Messing Computer Science & Communications
Gabe Darley Computer Science & Design

Advisor: **Dr. Aron Culotta**

Introduction

This project is a functional Frequently Asked Questions (FAQ) window, designed for a New Orleans nonprofit organization called *Families Helping Families* (FHF). The window receives a user question and transforms it into a discrete vector, which it can compare to other vector-questions set by the staff at FHF. It will return a question-answer pair most similar to the user's inquiry.

Problem statement

FHF has a small staff and serves a large population. The primary goal of this organization is to assist families in the Greater New Orleans area by "providing information and referral, training and education, and peer-to-peer support on issues related to disability." The ability of staff to communicate on an individual basis with clients is precious and limited. The organization requested a tool to mitigate client phone calls by answering the most basic questions regarding educational resources and legal processes on the website.

Methods

Before the window's final deployment, several tests were run that compared the efficacy of various **question-matching models**. In order to run these tests, the group developed several **paraphrases** of frequently asked questions to the test the ability of the window to match questions correctly. Once a model was chosen, the group went through some **fine-tuning** experimentation that optimized the success rate of the answer retriever.

Paraphrase Generation

The group was provided with 50 sample FAQ questions from FHF, from which 10 representative questions were chosen based on their unique syntax. For each of the 10 questions, 5 paraphrased "variations" were created organically, with changes to sentence order, word choice, and question length. Each time, the meaning of the question was preserved.

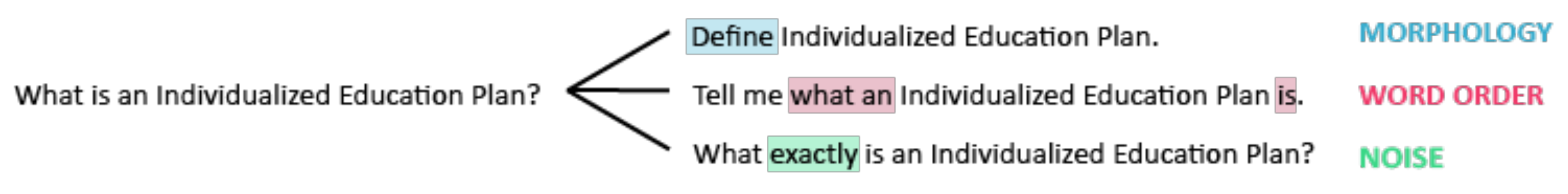

What is an Individualized Education Plan?
Define Individualized Education Plan.
Tell me what an Individualized Education Plan is.
What exactly is an Individualized Education Plan?
MORPHOLOGY
WORD ORDER
NOISE

Fig. 1 Sample paraphrase generation for a given question.

Language Models

In initial phases of the project, 4 unique language processing approaches were experimented with and tested:

- Binary Vectorization
- Term Frequency Inverse Document Frequency (TFIDF) Vectorization
- "SpaCy" Natural Language Processing Pipeline
- Bidirectional Encoder Representations from Transformers (BERT)

Each model was given the chance to match 50 paraphrases to their "correct" counterpart in the FAQ database. The efficacy of each model's ability to do so is represented in the chart below. It is important to note that each model utilized a vectorization process of some kind, calculating the cosine similarity between the user's vector and all database vectors to determine the best answer. It is the process of creating this vector that differed between models.

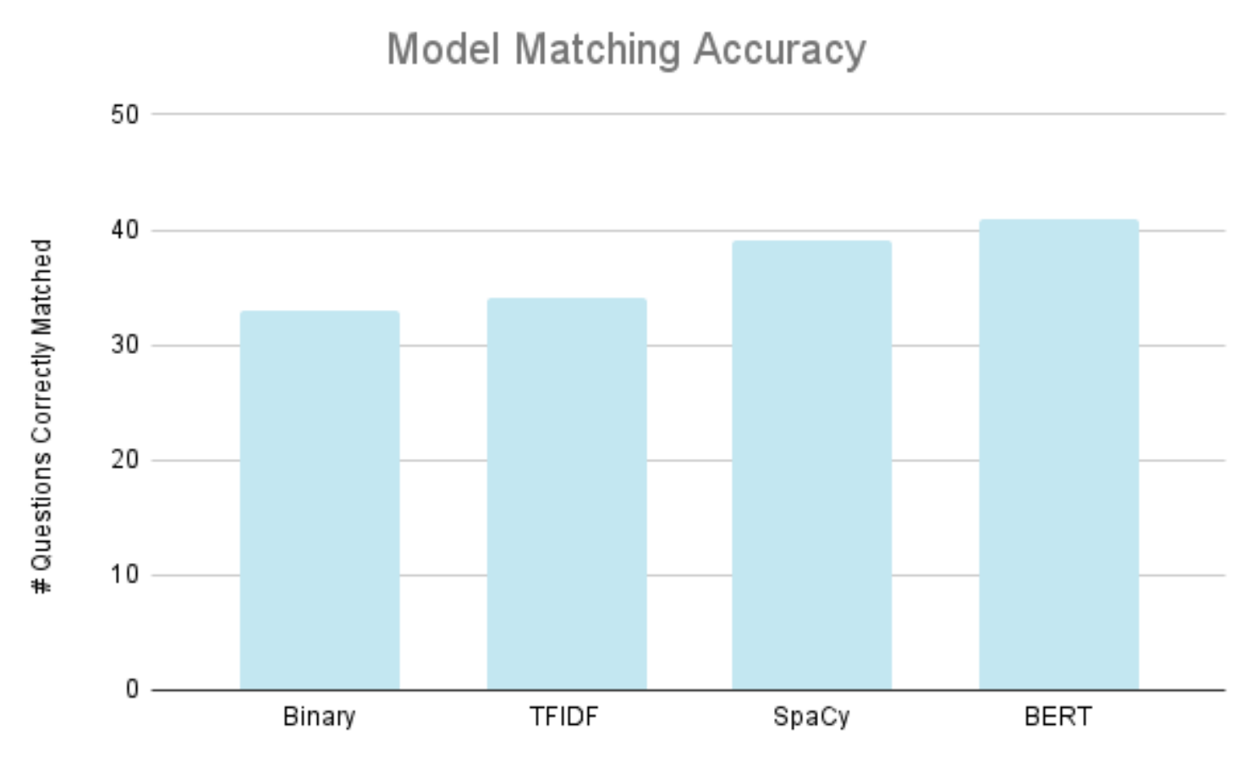


Fig. 2 Performance of various question matching models on 50 paraphrase scenarios.

Methods (cont.)

Upon analysis of the different models available, the group chose to build the window around the SpaCy model, despite its slight underperformance in accuracy in comparison to the BERT model. This was due to its high level of customizability and ease of use for future editors of the program after the group's departure from the project, as well as its compatibility with the deployment platform.

Fine-Tuning

The SpaCy model is a natural language processing "pipeline," which consists of multiple intermediate checkpoints at which a sentence is annotated and transformed. The "dependency parser", for example, will make note of the relationships between words, while the "lemmatizer" component will reduce certain words to their 'dictionary' form free of tense and inflection.

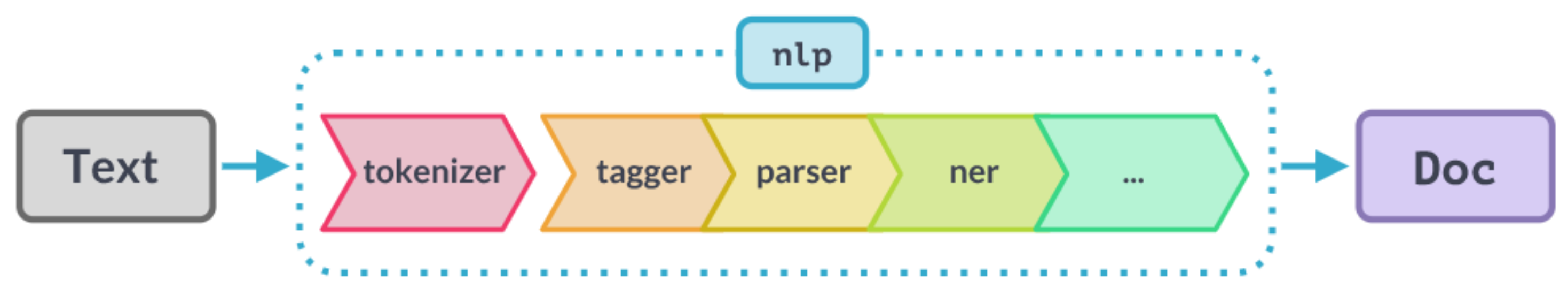


Fig. 3 Pipeline of natural language processing components, from SpaCy website.

The model relies on a "look-up table" of variable size, so each preset size was tested. Additionally, a custom "acronym expander" component was added by the team to improve accuracy in identifying common acronyms between questions.

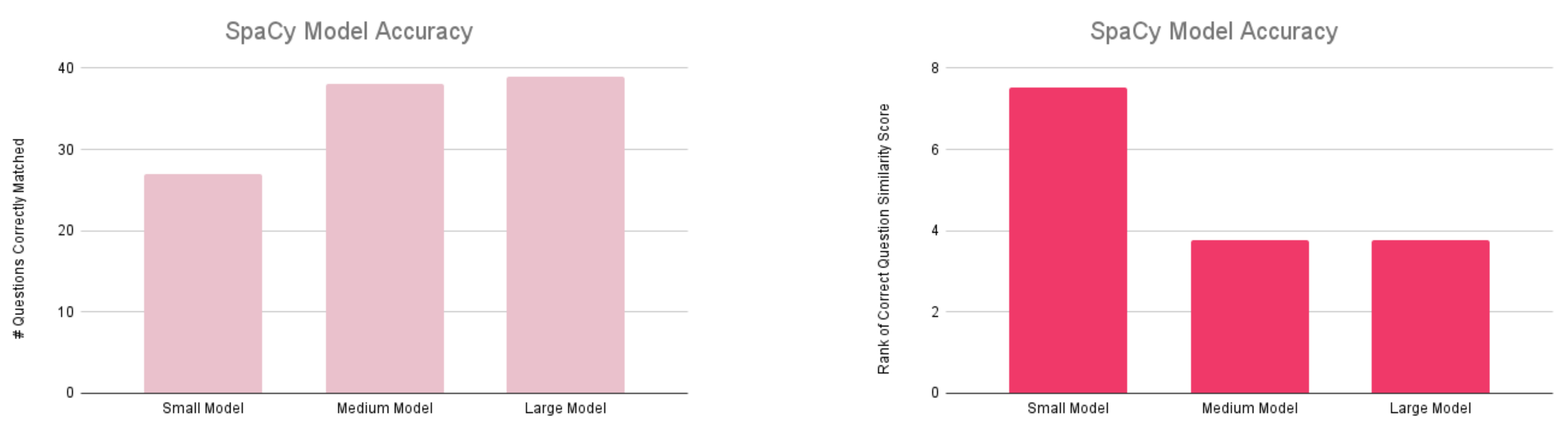


Fig. 4 Amount of correct question matches (left) and average relative similarity ranking of paraphrase to correct question (right) for different SpaCy model sizes.

Widget Structure

The make-up of the project's internal structure comes in two parts. The first part, consisting of JavaScript, HTML, and CSS handles user interaction and is the acting "front-end." The back-end component, where sentences are vectorized and information is stored, is hosted remotely on a server separate from the website. This allows for faster retrieval and reduces the burden and risk on the FHF site.

The back-end component stores FAQ pairs and acronym information locally for ease of access. When a user interacts with the window, a check is performed to see if the local file has been updated in the past 24 hours; if not, it rewrites the file by reading from an FHF-controlled spreadsheet.

Discussion

The intention of this project is not to completely solve a problem. Nonprofit employees are near ubiquitously overworked and under-resourced, and a coding project (likely) won't ever change that. The intention of this project also is not to replace the human conversations which make FHF so effective in the service of their community. This project is an application of existing technology to help an organization more efficiently conduct its day-to-day operations, and to leave more employee time allocated to the complex and essential conversations that arise for families of children with disabilities.

References

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

Vasiliev, Y. (2020). *Natural Language Processing with Python and SpaCy: A Practical Introduction*. No Starch Press.

Loper, E., & Bird, S. (2002). Nltk: The natural language toolkit.