Spacy Model and Tagging Data

Our named entity recognition model runs off the python library SpaCy. It's a supervised learning model, so we have to provide already-classified data for it to train on.

In Depth: ner_model.py

We start by loading in a model if we have it. That allows for recurrent training. If we don't have a model, we create a brand new one, using spacy's built in core english model.

However, that core model trains on different variables than we do (People, Organizations, etc vs Skills, Attributes). So, we create what's called a pipeline that includes the keywords we want it to learn.

Then, we run it through a loop and train it on the data that we've all tagged. We add the newly trained sections to the model, building it up with each piece of trained data.

Notes:

- tqdm in the for loop is creating that nice loading bar for us each time. It has no other purpose.
- I've set the dropout rate to 50%. That means that our neural network drops about half of its neurons. That's important for two reasons: 1) we currently don't have enough data and 2) the data we have is often repetitive. These two things might otherwise lead to overfitting, where the model "memorizes" the answers. By having a high dropout rate, we force it to actually learn based on context. That solves issues like it highlighting every "C" in the text as a programming language.

Finally, we have some validation going on that checks against a portion of our trained data that we've set aside. 20% of the data we tag goes into this section, and the model never sees it or gets to train on it until its time to run this validation.

Tagged Data

The single most important part of this process is making sure that our hand-tagged data is accurate and **consistent**. Currently, we tag data by providing a list of tuples for each paragraph, containing (start_index, end_index, tag_type) for each relevant word. There'll be more information on that in the data tagging tutorial.

One of the mini-projects available here would be to research some better methods for tagging. Whether that be better software, processes, or both, it would definitely come in handy!

Also - we could use some research on whether to keep the data tokenized or not when tagging. That might come in really handy when optimizing the model.

Potential Tasks

Priority	Difficulty	Description
5	2	**Method**: Devise a better method for tagging data, and present it to the group so that we can all consistently use the best practice. Requires some research and set-up, but shouldn't be too difficult.

Priority	Difficulty	Description
4	3	**Tokenization**: I would love to have feedback from a professor about whether to tokenize the data or not. This task would require either reading up on NLP studies or speaking with a professor, and relaying what best practices would look like for data processing.
3	4	**Validation**: Come up with a better validator for the model. I'm not sure if there are any issues with the current, but it is very simplistic. Again, would require looking into best practices but also coding up a testing suite for the model to be compared to.
2	5	**Tuning**: Play around with the hyperparameters of the model. Things like dropout rate, number of iterations, etc might all play a big role in squeezing out as much accuracy as we can. We'll never be able to tag tens of thousands of paragraphs, so this might be really crucial to getting an accurate prediction. You'll have to really dig into the weeds of Spacy to figure out how to interact with the more buried aspects of the model, but I can show you how hyperparameter tuning would work.
2	2?	**Logistics**: It would be a lot more efficient if we could use a high-powered machine to do things like scrape data quickly and train the model. Currently, this would take many hours otherwise. This task would be all about trying to secure a virtual machine from your respective school. Other p-ai groups have done it, so it is possible 😃.