SEASALT.AI ASR Challenge

A good machine learning engineer makes incremental improvements, but a *great* machine learning engineer finds ways to make breakthroughs. Here are a few general questions and open challenges for you to solve.

Our interview process includes:

1. a written challenge to demonstrate your attitude, thoughtfulness, ability (this one)
2. a coding interview to demonstrate your skills
3. an in-person interview with all the stakeholders: CEO, CTO, other teammates

General Questions

1. Are you looking for a full-time job or an internship?
2. What's the earliest date you could start full-time? If you can do part-time, what's the earliest date you could start on part-time?
3. Where do you see yourself in 5 years?

Application Challenge

4. Use dialogflow.com from Google or voiceflow.com to create a chatbot about yourself (you digital twin). The bot can enable the following kind of conversation:
    a. your bot: "Hello, I'm XXX's digital twin. I can answer a few questions about him/her".
    - interviewer: "Hi there, nice to meet you"

    b. your bot: "Hi, XXX said hi too!"
    - interviewer: "Can you tell me a bit about yourself?"

    c. your bot: "Sure, my name is XXX, I'm currently …"
    - interviewer: "Are you based in Seattle?"

    d. your bot: "Yes, I'm based on the eastside specifically/ No, I don't live in Seattle…"
    - conversation goes on…

    Record a video of the conversation **or send a link to your bot** for the interviewer to try.

Algorithm & System Design Challenge

5. Build a simple isolated word recognizer and improve the performance.
    a. Download the data here: https://drive.google.com/open?id=1CUSYUhn0vy-h4g-XVM9nxvpqUjkRpFfj
        i. **feneme_set.txt**: contains 306 five-character-long feneme names, one per line, resulting in 306 lines. Consider a feneme as a label for the feature vector.
        ii. **train_transcripts.txt**: is the "script" (words) that was read by the speakers whose speech comprises the training data. Each of the 53 words in the vocabulary were presented, in some randomized order, to speakers, and there were 10 such speakers, so that the script-file contains 10 blocks of 53 words, for a total of 530 lines of data.

iii. **train_features.txt**: contains the feature (feneme) sequences, one (long) feneme-string per line, corresponding to the utterance of each word in the transcript file described above. There are 530 lines.

iv. **train_endpoints.txt**: contains “end-point” information, or the information about the leading- and trailing-silence surrounding each utterance. This information is encoded in the form of two integers per line, say, i and j, to indicate that the last feneme of the leading silence is at position i, the first feneme of the trailing silence is at position j, and the speech corresponds to the (i + 1)-th through (j − 1)-th labels in the label-file. There are, again, 530 lines of data.

v. **test_features.txt**: contains the feature (feneme) sequences for the test set, one variable-length feneme label-string per line, corresponding to an utterance of each word in the test set. There are 530 lines

b. Train a recognizer using your favorite tools and models, which can properly predict words in train_transcripts.txt from feature vectors in train_features.txt.
c. Generate a list of words from feature vectors in test_features.txt.
d. Find ways to improve your recognizer, document them.
e. Suggestion: cut out a dev set form the training set, so you don’t overfit to the training data.

Write a detailed **report** of what you did in each step, including but not limited to tools you used, brief explanation of your implementation, different ways you tried to improve the model performance, as well as the word list you generated from test_features.txt using your models. Submit your **report** together with the **executables** for the interviewer to review.

Please submit your answers to https://forms.gle/XpcgT9QhWnNEumNy5 by 3/13/2020 9am.