INTERSPEECH 2019: Show & Tell Contribution
September 15–19, 2019, Graz, Austria

FarSpeech: Arabic Natural Language Processing for Live Arabic Speech

Mohamed Eldesouki, Naassih Gopee, Ahmed Ali, Kareem Darwish

Qatar Computing Research Institute
Hamad Bin Khalifa University, Doha, Qatar
`mohamohamed|ngopee|amali|kdarwish@hbku.edu.qa`

Abstract

This paper presents FarSpeech, QCRI's combined Arabic speech recognition, natural language processing (NLP), and dialect identification pipeline. It features modern web technologies to capture live audio, transcribes Arabic audio, NLP processes the transcripts, and identifies the dialect of the speaker. For transcription, we use QATS, which is a Kaldi-based ASR system that uses Time Delay Neural Networks (TDNN). For NLP, we use a SOTA Arabic NLP toolkit that employs various deep neural network and SVM based models. Finally, our dialect identification system uses multi-modality from both acoustic and linguistic input. FarSpeech[1] presents different screens to display the transcripts, text segmentation, part-of-speech tags, recognized named entities, diacritized text, and the identified dialect of the speech.

Index Terms: Speech Transcription, live speech recognition, Natural Language Processing

1. Introduction

Downstream processing of transcribed speech can enable a variety of applications such as information extraction and machine translation. In this paper, we present the FasSpeech web application, which couples QATS, our live Arabic Automatic Speech Recognition (ASR) system[1], and Farasa, our natural language processing pipeline [2], to perform a variety of NLP tasks on live speech. FarSpeech is designed to show more than the plain text speech transcripts. Farspeech has five different screens. The first four screens show the output of different Farasa NLP processors, namely; (1) Segmentation, which involves breaking words into its constituent prefix(es), stem, and suffix(es) and is essential for a variety of applications such as text retrieval and machine translation; (2) Part-of-Speech (POS) tagging, where we color code POS tags such as nouns, verbs, and adjectives using different colors; (3) Named-entity recognition (NER), which builds on segmentation and POS tagging to identify named entities that include persons, organizations, and locations; and (4) Text Diacritization, which involves the automatic recovery of short-vowels, a.k.a. diacritics, that are typically omitted in Arabic text, including in speech recognition output. ASR complicates NLP processing, because ASR output may contain recognition errors and the recognized text may contain a mix of Modern Standard Arabic (MSA) and dialects. Farasa is optimized for MSA. The final screen shows Arabic Dialect Identification (ADI) results, which attempts to label the input speech as either MSA, Egyptian, Gulf, North African, or Levantine. This is the first system that combines ASR, NLP and dialectic identification in a live streaming setup.

[1]`http://farspeech.qcri.org/`

2. System architecture

The Farspeech system is composed of four independent components, namely: the web application, the QATS ASR server, the Farasa NLP toolkit application server, and the dialect identification system. The complete system workflow diagram is shown in Figure 1. It performs the following steps:

- Capture input from a user's microphone through the interface.
- Spawn a worker that sends raw audio to the QATS ASR server.
- Get transcribed output from QATS and send it to Farasa web server.
- Retrieve Farasa results and display segmentation, POS tags, recognized named entities, and diacritized text.
- Send audio to dialect identification systems.
- Display output on Farspeech.

Since this is a live speech system, steps 1-6 are continually repeated as more audio input is received from the user. The data stream is continuously sent to the ASR server, and the incremental output is fetched to the Farasa web API for further processing allowing the different components to operate independently of each other.

2.1. The Web Application

The web application is comprised of a front-end and a back-end. The front-end presents the users with an interface to initialize their microphone and to record their audio input with the possibility of cancelling their session at any point. The front end is primarily built with the *Bootstrap*[2] framework. To handle speech we used *Wavesurfer*[3] and the *Dictate*[4] javascript library to enable microphone initialization and audio capture. *Dictate* converts the audio from WAV to raw binary audio file, which is then passed to the web application back-end that is built using the *Tornado Web Server*[5]. The Tornado application spawns workers which take care of each user's session. A worker passes the raw audio input to the ASR server and receives back the transcript as an output. Once the transcribed output is received, the data is sent to the Farasa web server through its web API, which returns processed text, including segmentation, POS tags, named entities, and diacritization. The raw audio together with its text transcripts is also sent in parallel to the dialect identification system which in turns return the mostly likely region from which the dialect originated. These steps are repeated for each partial audio input that the application receives from the user microphone input.

[2]`https://getbootstrap.com/`
[3]`https://wavesurfer-js.org/`
[4]`http://kaljurand.github.io/dictate.js/`
[5]`https://www.tornadoweb.org/`

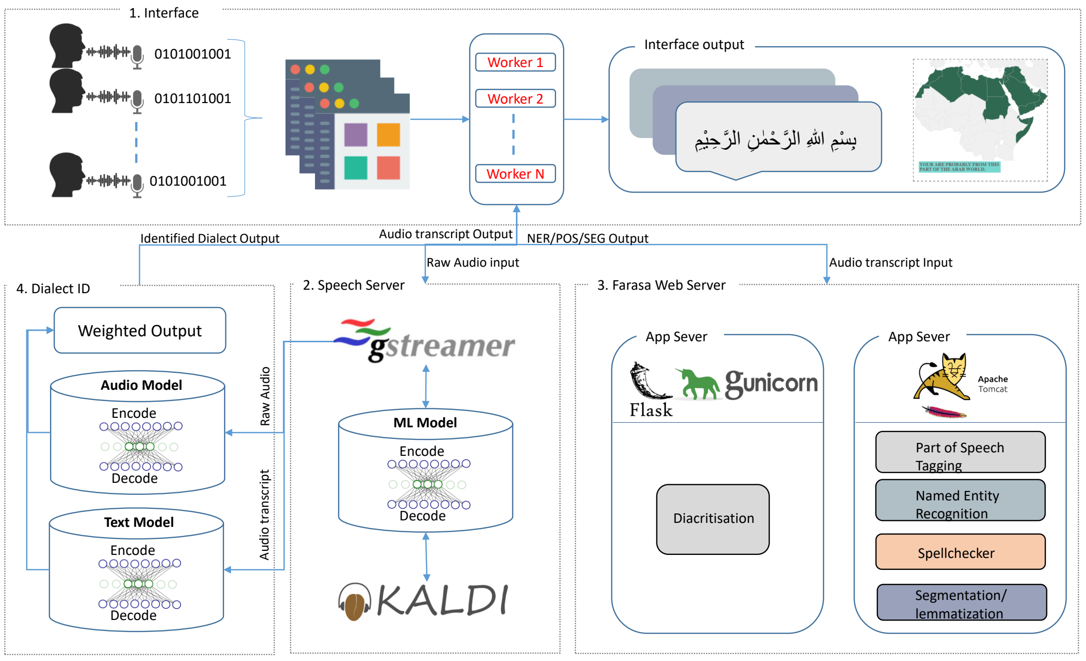

Figure 1: *FarSpeech System Overview*

2.2. Speech transcription

We use the QATS Speech-to-text transcription system that we built as part of QCRI's submission[3] to the 2016 Arabic Multi-Dialect Broadcast Media Recognition (MGB) Challenge[4]. The key features of the transcription system are as follows:
Acoustic Models: We experimented with various acoustic models; Time Delayed Neural Networks (TDNNs) [5], Long Short-Term Memory Recurrent Neural Networks (LSTM) and Bi-directional LSTM (BiLSTM) [6]. Though the performance of the BiLSTM acoustic model in terms of *Word Error Rate* is better than the TDNN, TDNN has a much better *real-time* factor while decoding. Therefore, we opted to use the TDNN acoustic model. More model details are available in Khurana et al. [3].
Language Model: We built a Kneser Ney smoothed trigram language model. The vocabulary size is restricted to the 100k most frequent words to improve the decoding speed and in-turn the *real-time factor* of the system. The choice of using a trigram model instead of a recurrent neural network model, as in the QATS offline system, was essential for keeping the decoding speed at a reasonable fast.

2.3. Natural language processing

For Arabic text language processing, we used our in-house Arabic NLP toolkit called Farasa[6] (meaning chivalry in Arabic). The pipeline includes segmentation, POS tagging, NER, a diacritization, spell checking, lemmatization, and dependency and syntactic parsing. Though Farasa is tuned for MSA, particularly for the news domain, it can handle other genres along with classical and dialectal Arabic, but at reduced accuracy. This is possible because of the large overlap between MSA and other varieties of Arabic. Farasa fills an important gap in the span of available tools. It is the only comprehensive suite of Arabic tools that is both open source and whose internal sub-components are competitive with the state of the art. In FarSpeech, we employ the segmenter, POS tagger, NER, and diacritizer.

[6]`http://farasa.qcri.org`

3. Conclusion

This paper presents FarSpeech, the QCRI system for live speech, NLP processing, and dialect identification. Currently, the system works very well for Arabic including frequent dialectal words. For future work, we aim to improve the system in several ways including having a tighter integration between ASR and NLP, and to extend its use to other applications such language learning.

4. References

[1] A. Ali, Y. Zhang, and S. Vogel, "Qcri advanced transcription system (qats)," in *Spoken Language Technology Workshop (SLT)*, 2014.

[2] A. Abdelali, K. Darwish, N. Durrani, and H. Mubarak, "Farasa: A fast and furious segmenter for arabic," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, 2016, pp. 11–16.

[3] S. Khurana and A. Ali, "QCRI advanced transcription system (QATS) for the arabic multi-dialect broadcast media recognition: MGB-2 challenge," in *Spoken Language Technology Workshop (SLT) 2016 IEEE*, 2016.

[4] A. Ali, P. Bell, J. Glass, Y. Messaoui, H. Mubarak, S. Renals, and Y. Zhang, "The mgb-2 challenge: Arabic multi-dialect broadcast media recognition," in *SLT*, 2016.

[5] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts." in *INTERSPEECH*, 2015, pp. 3214–3218.

[6] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." in *INTERSPEECH*, 2014.