Task based assistant core challenges: NLU

Robust language sensing

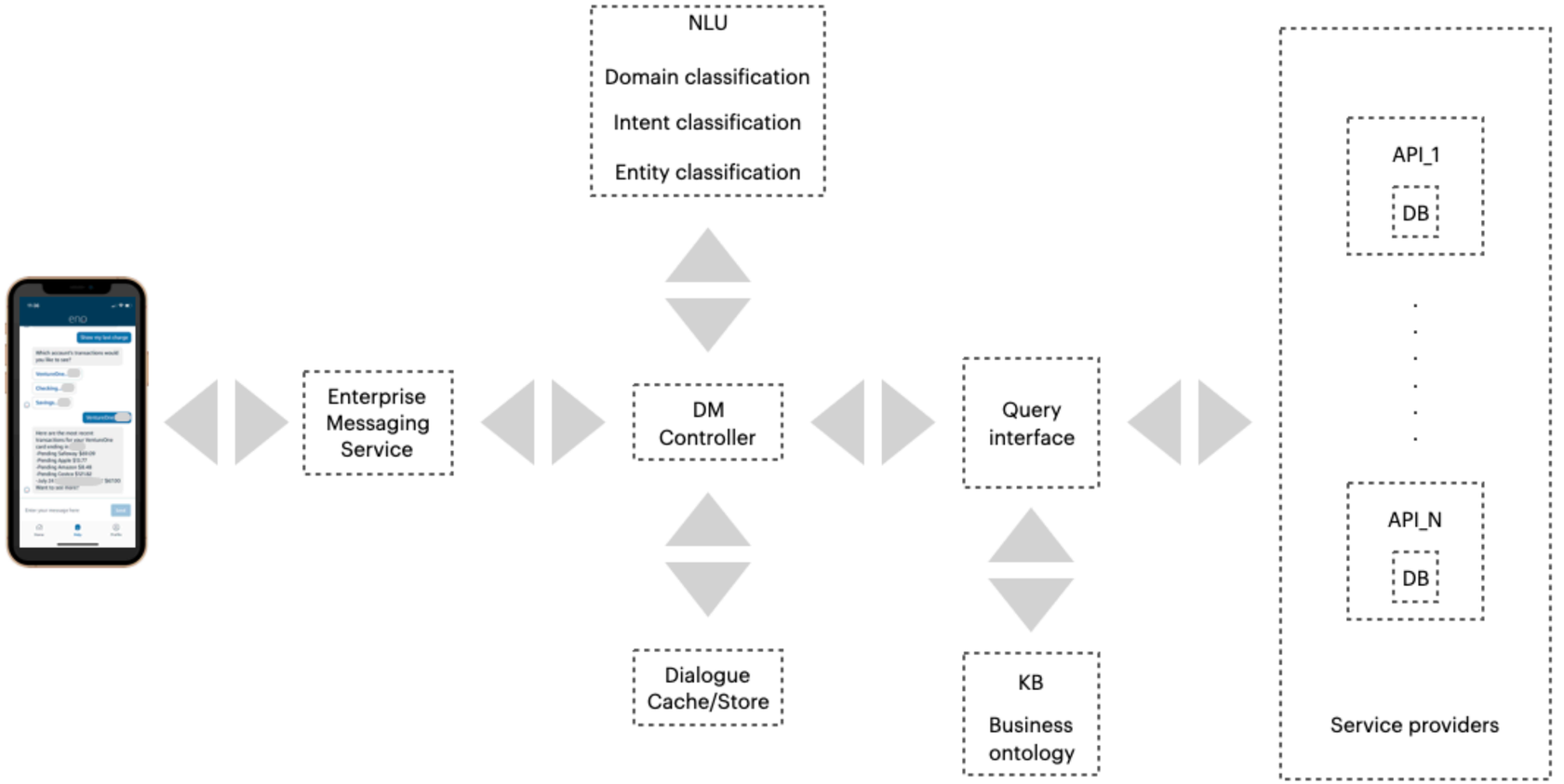
* **Spoken vs typed**
* **Slang vs formal**
* **Typos & dialects**
* **Microphone quality**
* **Intent, entity, domain rec.**
* **Implicit entities**

Dialogue state management

* Efficient slot filling
* Multi intent requests
* Contextual awareness

System scale

* Multi-domain support
* 3rd party integrations
* Performance



Generic dialogue system architecture

The voice - text gap

ASR systems nowadays are good, but not perfect. The quality of the language model scoring is often a limiting factor, especially when running on device.

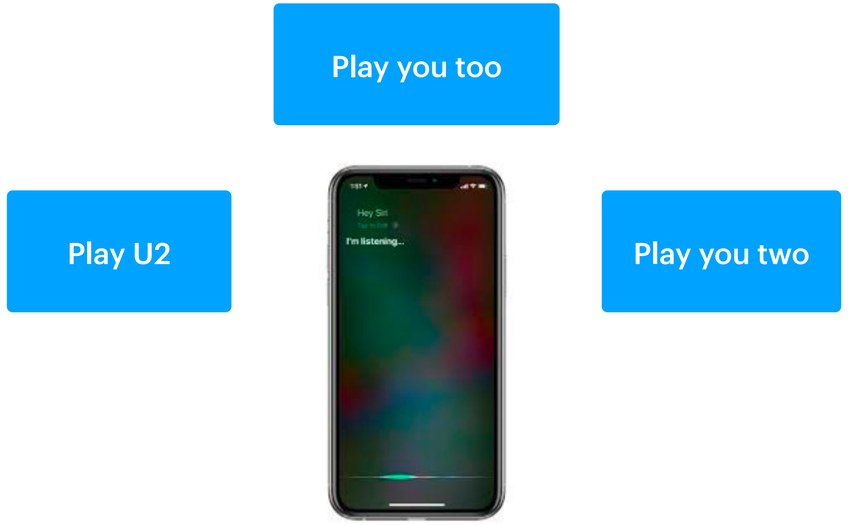
* Every domain / context / user can have a unique OOV set.

Written and spoken language are not generated from the same distribution!

Chatbot systems therefore need to perform language model refinement directly on the transcribed utterance

ML solutions

* Query expansion
  * Homophones / phonetic matching
  * Orthographic / grapheme matching
  * Semantic / synonym matching
* Transliteration



Telephonetic: Making Neural Language Models Robust to ASR and Semantic Noise

Chris Larson *Tarek Lahlou* *Diana Mingels* *Zachary Kulis* *Erik Mueller*

Capital One

{christopher.larson2, tarek.lahlou, diana.mingels, zachary.kulis, erik.mueller}@capitalone.com

Abstract

Speech processing systems rely on robust feature extraction to handle phonetic and semantic variations found in natural language. While techniques exist for desensitizing features to common noise patterns produced by Speech-to-Text (STT) and Text-to-Speech (TTS) systems, the question remains how to best leverage state-of-the-art language models (which capture rich semantic features, but are trained on only written text) on inputs with ASR errors. In this paper, we present `Telephonetic`, a data augmentation framework that helps robustify language model features to ASR corrupted inputs. To capture phonetic alterations, we employ a character-level language model trained using probabilistic masking. Phonetic augmentations are generated in two stages: a TTS encoder (Tacotron 2, WaveGlow) and a STT decoder (DeepSpeech). Similarly, semantic perturbations are produced by sampling from nearby words in an embedding space, which is computed using the BERT language model. Words are selected for augmentation according to a hierarchical grammar sampling strategy. Telephonetic is evaluated on the Penn Treebank (PTB) corpus, and demonstrates its effectiveness as a bootstrapping technique for transferring neural language models to the speech domain. Notably, our language model achieves a test perplexity of 37.49 on PTB, which to our knowledge is state-of-the-art among models trained only on PTB.

models such as BERT [2] and GPT-2 [3].

In this paper, we introduce the *Telephonetic* augmentation framework wherein data augmentations can be used to fine-tune language models on written text data so that they are better equipped to handle ASR errors. The term telephonetic is inspired by the popular game *telephone* wherein a message is passed sequentially and orally from person to person, aggregating phonetic and other errors along the way, until eventually the message has diverged significantly from its origin. This paper lays the foundation for generating similar errors by pairing neural speech synthesis systems with commodity ASR systems and reflecting the resulting errors into a training dataset. In Section 2 we discuss the character level language model evaluated in this paper as well as the companion training strategy. In Section 3 we present the core components of the telephonetic framework and provide experimental results and commentary in Section 4.

2. Character-based language modeling

The Language Model (LM) discussed in this paper builds upon the Char-CNN-LSTM architecture proposed in [6]. Rather than imposing causality to the learning task, as is commonplace with next-word prediction training, we instead use a masked LM training procedure inspired by Devlin *et al.* [2]. Letting $\mathbf{w}_i$ denote the i-th word in the text sequence $\mathbf{w} = [\mathbf{w}_0, \ldots, \mathbf{w}_{T-1}]$,