Noise Robust Named Entity Understanding for Voice Assistants

Deepak Muralidharan[*], **Joel Ruben Antony Moniz**[*], **Sida Gao**[*], **Xiao Yang**[*][†], **Justine Kao**, **Stephen Pulman**, **Atish Kothari**, **Ray Shen**, **Yinying Pan**, **Vivek Kaul**, **Mubarak Seyed Ibrahim**, **Gang Xiang**, **Nan Dun**, **Yidan Zhou**, **Andy O**, **Yuan Zhang**, **Pooja Chitkara**, **Xuan Wang**, **Alkesh Patel**, **Kushal Tayal**, **Roger Zheng**, **Peter Grasch**, **Jason D. Williams**, **Lin Li** [‡]

Apple

Abstract

Named Entity Recognition (NER) and Entity Linking (EL) play an essential role in voice assistant interaction, but are challenging due to the special difficulties associated with spoken user queries. In this paper, we propose a novel architecture that jointly solves the NER and EL tasks by combining them in a joint reranking module. We show that our proposed framework improves NER accuracy by up to 3.13% and EL accuracy by up to 3.6% in F1 score. The features used also lead to better accuracies in other natural language understanding tasks, such as domain classification and semantic parsing.

1 Introduction

Understanding named entities correctly when interacting with virtual assistants (e.g. "Call Jon", "Play Adele hello", "Score for Warrior Kings game") is crucial for a satisfying user experience. However, NER and EL methods that work well on written text often perform poorly in such applications: utterances are relatively short (with just 5 tokens, on average), so there is not much context to help disambiguate; speech recognizers make errors ("Play Bohemian raspberry" for "Play Bohemian Rhapsody"); users also make mistakes ("Cristiano Nando" for "Cristiano Ronaldo"); non-canonical forms of names are frequent ("Shaq" for "Shaquille O'Neal"); and users often mention new entities unknown to the system.

In order to address these issues we propose a novel Named Entity Understanding (NEU) system that combines and optimizes NER and EL for noisy spoken natural language utterances. We pass multiple NER hypotheses to EL for reranking, enabling NER to benefit from EL by including information from the knowledge base (KB).

We also design a retrieval engine tuned for spoken utterances for retrieving candidates from the KB. The retrieval engine, along with other techniques devised to address fuzzy entity mentions, lets the EL model be more robust to partial mentions, variation in named entities, use of aliases, as well as human and speech transcription errors.

Finally, we demonstrate that our framework can also empower other natural language understanding tasks, such as domain classification (a sentence classification task) and semantic parsing.

2 Related Work

There have been a few attempts to explore NER on the output of a speech pipeline (Ghannay et al., 2018; Abujabal and Gaspers, 2018; Coucke et al., 2018). Among these, our NER model is closest to Abujabal and Gaspers (2018) and Coucke et al. (2018); however, unlike the former, we use a richer set of features rather than phonemes as input, and unlike the latter, we are able to use a deep model because of the large volume of data available.

EL has been well explored in the context of clean (Martins et al., 2019; Kolitsas et al., 2018; Luo et al., 2015) and noisy text inputs (Eshel et al., 2017; Guo et al., 2013; Liu et al., 2013), but as with NER, there have been only a few efforts to explore EL in the context of transcribed speech (Benton and Dredze, 2015; Gao et al., 2017), although crucially, both these works assume gold standard NER and focus purely on the EL component.

Traditionally, a pipelined architecture of NER followed by EL has been used to address the entity linking task (Lin et al., 2012; Derczynski et al., 2015; Bontcheva et al., 2017; Bowden et al., 2018). Since these approaches rely only on the best NER hypothesis, errors from NER propagate to the EL step. To alleviate this, joint models have been proposed: Sil and Yates (2013) proposed an NER+EL model which re-ranks candidate mentions and entity links produced by their base model. Our work

[*] **Equal contributions.**
[†] graceyx.scut@gmail.com
[‡] lli9@apple.com

differs in that we use a high precision NER system, while they use a large number of heuristically obtained Noun Phrase (NP) chunks and word n-grams as input to the EL stage. Luo et al. (2015) jointly train an NER and EL system using a probabilistic graphical model. However, these systems are trained and tested on clean text and do not address the noise problems we are concerned with.

3 Architecture Design

For a given utterance, we first detect and label entities using the NER model and generate the top-l candidate hypotheses using beam search. The EL model consists of two stages: (i) candidate retrieval and (ii) joint linking and re-ranking. In the retrieval stage, for each NER hypothesis, we construct a structured search query and retrieve the top-c candidates from the retrieval engine. In the ranking stage, we use a neural network to rank these candidate entity links within each NER hypothesis while simultaneously using rich signals (entity popularity, similarity between entity embeddings, the relation across multiple entities in one utterance, etc.) from these entity links as additional features to re-rank the NER hypotheses from the previous step, thus jointly addressing both the NER and EL tasks.

3.1 NER

For the NER task, following Lample et al. (2016) we use a combination of character and word level features. They are extracted by a bi-directional LSTM (biLSTM) (Hochreiter and Schmidhuber, 1997), and then concatenated with pre-trained GloVe word embeddings [1] (Pennington et al., 2014) to pass through another biLSTM and fed into a CRF model to produce the final label prediction based on a score $s(\tilde{\mathbf{y}}_\mathbf{i}, \mathbf{x}; \theta)$ that jointly optimizes the probability of labels for the tokens and the transition score for the entire sequence $\tilde{\mathbf{y}}_\mathbf{i} = (y_1, \ldots, y_T)$ given the input $\mathbf{x}$:

$$s(\tilde{\mathbf{y}}_\mathbf{i}, \mathbf{x}; \theta) = \sum_{t=0}^{T} \left(\psi_{t,\theta}(y_t) + \phi_{t,t+1}(y_t, y_{t+1})\right),$$

where $\psi_{t,\theta}$ is the biLSTM prediction score from the label y_t of the t^{th} token, and $\phi(j,k)$ is the transition score from label j to label k.

During training, we maximize the probability of the correct label sequence p_{seq}, which is defined as

$$p_{seq}(\tilde{\mathbf{y}}_\mathbf{i}, \mathbf{x}; \theta) = \frac{\exp(s(\tilde{\mathbf{y}}_\mathbf{i}, \mathbf{x}; \theta))}{\sum_{\tilde{\mathbf{y}}_\mathbf{j} \in S} \exp\left(s(\tilde{\mathbf{y}}_\mathbf{j}, \mathbf{x}; \theta)\right)},$$

where $\tilde{\mathbf{y}}_\mathbf{i}$ is the label sequence for hypothesis i, and S is the set of all possible label sequences.

During inference, we generate up to 5 NER alternatives for each utterance using beam search. We also calculate a mention level confidence p_{men} for each entity mention $\mathbf{m_k}$. p_{men} is computed by aggregating the sequence level confidence for all the prediction sequences that share the same mention sub-path $\mathbf{m_k}$:

$$p_{men}(\mathbf{m_k}, \mathbf{x}; \theta) = \frac{\sum_{\tilde{\mathbf{y}}_\mathbf{i} \in S_{\mathbf{m_i}}} \exp(s(\tilde{\mathbf{y}}_\mathbf{i}, \mathbf{x}; \theta))}{\sum_{\tilde{\mathbf{y}}_\mathbf{j} \in S} \exp(s(\tilde{\mathbf{y}}_\mathbf{j}, \mathbf{x}; \theta))},$$

where S_{m_i} is the set of prediction sequences that all have $\mathbf{m_k}$ as the prediction for the corresponding tokens. Both p_{seq} and p_{men} are computed by dynamic programming, and serve as informative features in the EL model.

3.2 Joint Linking and Re-ranking

The entity linking system follows the NER model and consists of two steps: candidate retrieval, and joint linking and re-ranking.

To build the candidate retrieval engine, we first index the list of entities in our knowledge base, which can be updated daily to capture new entities and change of their popularity. To construct the index, we iterate through the flattened list of entities and construct token-level unigram, bigram and trigram terms from the surface form of each entity. Apart from using the original entity names, we also use common aliases, harvested from usage logs, for popular entities (e.g. LOTR as an alias for "Lord of the Rings") to make the retrieval engine more robust to commonly occurring variations. Next, we create an inverted index which maps the unique list of n-gram terms to the list of entities that these n-grams are part of, also known as posting lists. Further, to capture cross-entity relationships in the knowledge base (such as relationships between an artist and a song or two sports teams belonging to the same league), we assign a pointer[2] for each

[1]We also tried more recent contextual embeddings such as BERT (Devlin et al., 2019), and empirically observed very little difference in performance when compared to GloVe. So we adopt GloVE, which is substantially more efficient in terms of inference time required.

[2]Each entity in our knowledge base consists of metadata (for example, a song entry in our knowledge base would contain metadata such as the music artist, album, year the song was released in etc.) that we leverage to automatically construct these relationship pointers.

entity in the knowledge base to its related entities and this relational information is leveraged by the EL model for entity disambiguation (described in 5.2). We then compute the tf-idf score for all the n-gram terms present in the entities and store them in the inverted index.

For each hypothesis predicted by the NER model we query the retrieval engine with the corresponding text. We first send the query through a high-precision seq-to-seq correction model (Schmaltz et al., 2017; Ge et al., 2019) trained using common errors observed in usage. Next, we construct n-gram features from the corrected query in a similar way to the indexing phase and retrieve all entities matching these n-gram features in our inverted index. Additionally, we use synonyms derived from usage for each term in the query to expand our search criteria: for example, our synonym list for “Friend" contains “Friends", which matches the TV show name which would have been missed if only the original term was used.

For each entity retrieved, we get the tf-idf score for the terms present in the query chunk from the inverted index. We then aggregate the tf-idf scores of all the terms present in the query for this entity and linearly combine this aggregate score with other attributes such as popularity (i.e. prior usage probability) of the entity to generate a final score for all retrieved entity candidates for this query. Finally, we perform an efficient sort across all the entity candidates based on this score and return a top-c (in our case $c = 25$) list filtered by the entity type detected by the NER model for that hypothesis. These entity candidates coupled with the original NER hypothesis are sent to the ranker model described below for joint linking and re-ranking.

Following the candidate retrieval step, we introduce a neural model to rerank the candidate entities, aggregating features from both the NER model and the candidate retrieval engine.

The EL model scores each entity linking hypothesis separately. An entity linking hypothesis consists of a prediction from the NER model (which consists of named entity chunks in the input utterance and their types), and the candidate retrieval results for each chunk. Formally, we define an entity linking hypothesis $\mathbf{y}$ with k entity predictions as:

$$\mathbf{y} = \{\mathbf{f}_{\text{utter}}, \mathbf{f}_{\text{NER}}, \mathbf{f}_{\text{CR}}, \{j \in \{1 \ldots k\} : (\mathbf{m}_j, \mathbf{e}_j)\}\}$$

where $\mathbf{m}_j$ is the j-th mention in the utterance, and $\mathbf{e}_j$ is the entity name associated with this mention from the knowledge base. $\mathbf{f}_{\text{utter}}$, $\mathbf{f}_{\text{NER}}$, $\mathbf{f}_{\text{CR}}$ are features derived from the original utterance text, the NER model and the candidate retrieval system respectively. In our system, $\mathbf{f}_{\text{utter}}$ is a representation of the utterance from averaging the pre-trained word embeddings for the tokens in the utterance. Intuitively, having a dense representation of the full utterance can help the EL model better leverage signals from the utterance context. $\mathbf{f}_{\text{NER}}$ includes the type of each mention, as well as the sequence and mention confidence computed by the NER model. $\mathbf{f}_{\text{CR}}$ includes popularity, and whether a relation exists between the retrieved entities in $\mathbf{y}$.

To be robust to noise, the EL model adopts a pair of CNNs to compare each entity mention $\mathbf{m}_j$ and its corresponding knowledge base entity name $\mathbf{e}_j$. The CNN learns a name embedding with one-dimensional convolution on the character sequence, and the kernel parameters are shared between the CNN used for user mention and the one used for the canonical name. A character-based text representation model is better at handling mis-transcriptions or mis-pronounced entity names. While a noisy entity name may be far from the canonical name in the word embedding space when they are semantically different, they are usually close to each other in the character embedding space due to similar spellings. To model the similarity between CNN name embeddings of $\mathbf{m}_j$ and $\mathbf{e}_j$, we use the standard cosine similarity as a baseline, we experiment with an MLP that takes the concatenated name embeddings as input. We are able to model more expressive interactions between the two name embeddings with the MLP, and in turn better handle errors. Finally, we concatenate the similarity features with other features as input to another MLP that computes the final score for $\mathbf{y}$. Formally, the scoring function is defined in Equation 1, where $\oplus$ means concatenation.

$$s(\mathbf{y}) = \mathbf{MLP}(\mathbf{f}_{\text{utter}} \oplus \mathbf{f}_{\text{NER}} \oplus \mathbf{f}_{\text{CR}} \bigoplus_{j=1}^{k} [\mathbf{MLP}(\mathbf{CNN}(\mathbf{m}_j), \mathbf{CNN}(\mathbf{e}_j)) \oplus \mathbf{CNN}(\mathbf{m}_j) \oplus \mathbf{CNN}(\mathbf{e}_j)]) \quad (1)$$

In our data, the number of entity mentions in an utterance averages less than 3. We pad the entity feature sequence to length 5, which provides a good coverage. In the scoring model above, we use a simple concatenation to aggregate the embedding similarities of multiple entity mentions which empirically performs as well as sequence models like LSTM, while being much cheaper in computation.

To train the EL model, we use the standard max-margin loss for ranking tasks. If for the i-th example, we denote the ground truth as $\mathbf{y^*}_i$ and an incorrect prediction as $\hat{\mathbf{y}}_i$, and the scoring function $s(\cdot)$ is as defined in Equation 1, the loss function is

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}[\gamma(\hat{\mathbf{y}}_i, \mathbf{y^*}_i) + s(\hat{\mathbf{y}}_i) - s(\mathbf{y^*}_i)]_+. \quad (2)$$

The max-margin loss encourages the ground truth score to be at least a margin γ higher than the score of an incorrect prediction. The margin is defined as a function of the ground truth and the incorrect prediction, thus adaptive to the quality of prediction. A larger margin is needed when the incorrect prediction is further away from the ground truth. For our reranking task, we set a smaller margin when only the resolved entities are incorrect but the NER result is correct, and a larger margin when the NER result is wrong. This adaptive margin helps rerank NER hypotheses even when the model cannot rank the linking results correctly. During training, we uniformly sample the negative predictions from the candidates retrieved by the retrieval engine.

3.3 Improvement on Other Language Understanding Tasks

We also explore the impact of our NEU feature encoding on two tasks: a domain classifier and a domain-specific shallow semantic parser.

3.3.1 Domain Classification

Domain classification identifies which domain a user's request falls into: sports, weather, music, etc., and is usually done by posing the task as sequence classification: our baseline uses word embeddings and gazetteer features as inputs to an RNN, in a manner similar to Chen et al. (2019).

Consider a specific token t. Let a be the number of alternatives used from the NER model in the domain classifier (which we treat as a hyperparameter), p_i represent the (scalar) sequence level confidence score $p_{seq}(\tilde{\mathbf{y}}_\mathbf{i}, \mathbf{x}; \theta)$ of the i^{th} NER alternative defined in Section 3.1, c_i represent an integer for the entity type that NER hypothesis i assigns to the token t, and $\mathbf{o}(.)$ represent a function converting an integer into its corresponding one-hot vector. Then the additional NER feature vector $\mathbf{f_r}$ concatenated to the input vector fed into token t as part of the domain classifier can be written as:

$$\mathbf{f_r} = \bigoplus_{i=1}^{i=a} p_i \mathbf{o}(c_i). \quad (3)$$

Likewise, for the featurization that uses both NER and EL, let a be the number of alternatives used from the NER+EL system in the domain classifier (also a hyperparameter); these a alternatives are now sorted by the scores from the EL hypotheses, as opposed to the sequence level confidence scores from NER. Let s_i be the i^{th} re-ranked alternative's cosine similarity score between the mention and knowledge base entity name as output by the EL model. p_i and c_i are consistent with our earlier notation, except that they now correspond to the i^{th} NER alternative after re-ranking. Then the additional NER+EL feature vector $\mathbf{f_u}$ concatenated to the input fed into token t as part of the domain classifier can be written as:

$$\mathbf{f_u} = \bigoplus_{i=1}^{i=a} p_i \mathbf{o}(c_i) \oplus s_i \mathbf{o}(c_i). \quad (4)$$

3.3.2 Semantic Parsing

Our virtual assistant also uses domain-specific shallow semantic parsers, running after domain classification, responsible both for identifying the correct intent that the user expects (such as the "play" intent associated with a song) and for assigning semantic labels to each of the tokens in a user's utterance (such as the word "score" and "game" respectively being tagged as tokens related to a sports statistic and sports event respectively in the utterance "What's the score of yesterday's Warriors game?"). Each semantic parser is structured as a multi-task sequence classification (for the intent) and sequence tagging (for the token-level semantic labelling) task, with our production baseline using word embeddings and gazetteer features as inputs into an RNN similar to our domain classifier. Here, $\mathbf{f_r}$ and $\mathbf{f_u}$ are featurized as described above. Note that in contrast to the NEU system, the semantic parser uses a domain-specific ontology, to enable each domain to work independently and to not be encumbered by the need to align ontologies.

4 Datasets and Training Methodology

To create our datasets, we randomly sampled around $600k$ unique anonymous English transcripts (machine transcribed utterances), and annotated them with NER and EL labels. Utterances are subject to Apple's baseline privacy practices with respect to Siri requests, including that such requests are not associated with a user's Apple ID, email address, or other data Apple may have from a user's use of other Apple services, and have been filtered as described in Section 7. We then split the annotated data into 80/10/10 for train, development and test sets. For both the NER and EL tasks, we report our results on test sets sampled from the "music", "sports" and "movie & TV" domains. These are popular domains in the usage and have a high percentage of named entities: with an average of 0.6, 1.1 and 0.7 entities for each utterance in the 3 domains respectively. To evaluate model performance specifically on noisy user inputs, we select queries from the test sets that are marked as containing speech transcription or user errors by the annotators and report results on this "noisy" subset, which constitutes 13.5%, 12.7% data for movie&TV and music domain respectively when an entity exists. [3] To evaluate the relation feature, we also look at the "related" subset where a valid relation exists in the utterance. This subset consists 13.4% and 5.3% of data for the music and sports domain with at least one entity. [4]

We first train the NER model described in Section 3.1. Next, for every example in our training dataset, we run inference on the trained NER model and generate the top-5 NER hypotheses using beam search. Following this, we retrieve the top 25 candidates for each of these hypotheses using our search engine combined with the ground truth NER and EL labels and fed to the EL model for training.

To measure the NER model performance, we use the standard NER F1 metric used for the CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003). To measure the quality of the top-5 NER hypotheses, we compute the oracle top-5 F1 score by comparing and choosing the best alternative hypothesis among the 5 and calculate its F1 score, for each test utterance. In this manner, we also know the upper bound that EL can reach from reranking NER hypotheses. As described in section 3.2, the EL model is optimized to perform two tasks simultaneously: entity linking and reranking of NER hypotheses. Hence to evaluate the performance of the EL model, we use two metrics: reranked NER-F1 score and the EL-F1 score. The reranked NER F1 score is measured on the NER predictions according to the top EL hypothesis, and is defined in the same way as the previous NER task. To evaluate entity linking quality, we adopt a strict F1 metric similar to the one used for NER. Besides entity boundary and entity type, the resolved entity also needs to be correct for the entity prediction to be counted as a true positive.

For NER model training, we use standard mini-batch gradient descent using the Adam optimizer with an initial learning rate of 0.001, a scheduled learning rate decay of 0.99, LSTM with a hidden layer of size 350 and a batch size of 256. We apply a dropout of 0.5 to the embedding and biLSTM layers, and include token level gazetteer features (Ratinov and Roth, 2009) to boost performance in recognizing common entities. We linearly project these gazetteer features and concatenate the projection with the 200 dimensional word embeddings and 100 dimensional character embeddings which are then fed into the biLSTM followed by the CRF.

For EL, the character CNN model we use has two layers, each with 100 convolution kernels of size 3, 4, and 5. Character embedding are 25 dimensional and trained end to end with the entity linking task. The MLP for embedding similarity takes the concatenation of two name embeddings, as well as their element-wise sum, difference, minimum, maximum, and multiplication. It has two hidden layers of size 1024 and 256, with output dimension 64. Similarity features of mentions in the prediction are averaged, while the other features like NER confidence and entity popularity are concatenated to the representation. The final MLP for scoring has two hidden layers, with size 256 and 64. We train the model on 4 GPUs with synchronous SGD, and for each gradient step we send a batch of 100 examples to each GPU.

5 System Evaluation

5.1 Results

We present F1 scores in different domains of the NER and EL model in Table 1. Since the EL model takes 5 NER hypotheses as input, it also acts as a reranker of the NER model, and we show substantial

[3] Sports domain does not have the annotation for noisy data available when this experiment was conducted.

[4] Our KB does not have relation information for movie&TV domain.

improvements on top-1 NER F1 score consistently over all test sets.

	NER F1 top-1/top-5	reranked NER F1	EL F1
movie&TV	78.76 / 96.83	81.62	79.67
music	84.27 / 97.26	87.40	84.95
sports	92.97 / 99.15	93.48	91.13

Table 1: Results for the best model setting. NER F1 are reported on the top-1 and top-5 NER prediction from the NER model that provides features for EL. Reranked NER F1 and EL F1 are reported on top-1 prediction from the best EL model selected by development sets.

In Table 2, we show improvements achieved by several specific model design choices and features on entity linking performance. Table 2(a) shows the MLP similarity substantially improves entity linking accuracy with its capacity to model text variations, especially on utterances with noisy entity mentions. The relation feature is powerful for disambiguating entities with similar names, and we show a considerable improvement in EL F1 on the subset of utterances that have related entities in Table 2(b). Table 2(c) shows utterance embeddings brought improvements in the music, and media & TV domains. The improvement brought by log-scale popularity feature is the largest for the movie & TV domain as shown in Table 2(d), where the popularity distribution has extremely long tails compared to other domains.

5.2 Qualitative Analysis

We provide a few examples to showcase the effectiveness of our NEU system. Firstly, the EL model is able to link noisy entity mentions to the corresponding entity canonical name in the knowledge base. For instance, when the transcribed utterance is "play Carla Cabello", the EL model is able to resolve the mention "Carla Carbello" to the correct artist name "Camila Cabello".

Secondly, the EL model is able to recover from errors made by the NER system by leveraging the knowledge base to disambiguate entity mentions. The reranking is especially powerful when the utterance contains little context of the entity for the NER model to leverage. For example, for "Doctor Strange", the top NER hypothesis labels the full utterance as a generic "Person" type, and after reranking, EL model is able to leverage the popularity information ("Doctor Strange" is a movie that was recently released and has a high popularity in our knowledge base) and correctly label the utterance as "movieTitle". Reranking is also effective when the entity mentions are noisy, which will cause mismatches for the gazetteer features that NER uses. For "play Avengers Age of Ultra", the top NER hypothesis only predicts "Avengers" as "movieTitle", while after reranking, the EL model is able to recover the full span "Avengers Age of Ultra" as a "movieTitle", and resolve it to "Avengers: Age of Ultron", the correct canonical title.

The entity relations from the knowledge base are helpful for entity disambiguation. When the user refers to a sports team with the name "Giants", they could be asking for either "New York Giants", a National Football League (NFL) team, or "San Francisco Giants", a Major League Baseball team. When there are multiple sports team mentions in an utterance, the EL model leverages a relation feature from the knowledge base indicating whether the teams are from the same sports league (as the user is more likely to mention two teams from the same league and the same sport). Knowing entity relations, the EL model is able to link the mention "Giants" in "Cowboys versus Giants" to the NFL team, knowing that "Cowboys" is referring to "Dallas Cowboys".

To validate the utility of our proposed NEU framework, we illustrate performance improvements in the Domain Classifier and the Semantic Parsers corresponding to the three domains (music, movies & TV and sports) as described in Section 3.3. Table 3 reports the classification accuracy for the Domain Classifier and the parse accuracies for the Semantic Parsers (the model is said to have predicted the parse correctly if all the tokens are tagged with their correct semantic parse labels). We observe substantial improvements in all 4 cases when NER features are used as additional input, given all the other components of the system being the same. In turn, we observe further improvements when our NER+EL featurization is used.

6 Conclusion

In this work, we have proposed a Named Entity Understanding framework that jointly identifies and resolves entities present in an utterance when a user interacts with a voice assistant. Our proposed architecture consists of two modules: NER and EL, with the EL serving the additional task of possibly correcting the recognized entities from NER

	(a) + MLP
movie&TV	+3.58
(noisy)	+9.67
music	+2.05
(noisy)	+10.03

	(b) + Relation
music	+0.86
(related)	+1.97
sports	+0.07
(related)	+0.81

	(c) + Utterance Embedding	(d) + Log-scale Popularity
movie&TV	+0.25	+0.27
music	+0.39	+0.02
sports	-0.07	+0.08

Table 2: EL mean F1 relative % improvements, reported on 10 runs average.

	A	B	C
DC	88.95	89.46	**90.04**
SP [movie&TV]	89.62	90.99	**91.67**
SP [music]	83.97	84.26	**84.42**
SP [sports]	86.37	**86.47**	86.46

Table 3: Results for domain classifier (first row) and semantic parser. A is the baseline, B is A+NER, C is A+NER+EL.

by leveraging rich signals from entity links in the knowledge base while simultaneously linking these entities to the knowledge base. With several design strategies in our system targeted towards noisy natural language utterances, we have shown that our framework is robust to speech transcription and user errors that occur frequently in spoken dialog systems. We have also shown that featurizing the output of NEU and feeding these features into other language understanding tasks substantially improves the accuracy of these models.

7 Ethical Considerations

We randomly sampled transcripts from Siri production datasets over a period of months, and we believe it to be a representative sample of usage in the domains described. In accordance with Apple's privacy practices with respect to Siri requests, Siri utterances are not associated with a user's Apple ID, email address, or other data Apple may have from a user's use of other Apple services. In addition to Siri's baseline privacy guarantees, we filtered the sampled utterances to remove transcripts that were too long, contained rare words, or contained references to contacts before providing the dataset to our annotators.

Acknowledgements

We would like to thank Alex Acero, Anmol Walia, Arjun Rangarajan, Barry-John Theobald, Bhagyashree Shekawat, Christopher Klein, Dhivya Piraviperumal, Hong Yu, Jianpeng Cheng, Jiarui Lu, John Giannandrea, John Keesling, Katy Linsky, Lanqing Wang, Ryan Templin, Robert Daland, Shruti Bhargava, Xi Chen and Yvonne Xiao for discussions and for their help and feedback. We also want to thank all the anonymous reviewers for the helpful feedback and suggestions on this work.

References

Abdalghani Abujabal and Judith Gaspers. 2018. Neural named entity recognition from subword units. *arXiv preprint arXiv:1808.07364*.

Adrian Benton and Mark Dredze. 2015. Entity linking for spoken language. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 225–230, Denver, Colorado. Association for Computational Linguistics.

Kalina Bontcheva, Leon Derczynski, and Ian Roberts. 2017. Crowdsourcing named entity recognition and entity linking corpora. In *Handbook of Linguistic Annotation*, pages 875–892. Springer.

Kevin Bowden, Jiaqi Wu, Shereen Oraby, Amita Misra, and Marilyn Walker. 2018. SlugNERDS: A named entity recognition tool for open domain dialogue systems. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Xi C Chen, Adithya Sagar, Justine T Kao, Tony Y Li, Christopher Klein, Stephen Pulman, Ashish Garg, and Jason D Williams. 2019. Active learning for domain classification in a commercial spoken personal assistant. *Proc. Interspeech 2019*, pages 1478–1482.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke Van Erp, Genevieve Gorrell, Raphaël

Troncy, Johann Petrak, and Kalina Bontcheva. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named entity disambiguation for noisy text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 58–68, Vancouver, Canada. Association for Computational Linguistics.

Ning Gao, Douglas W. Oard, and Mark Dredze. 2017. Support for interactive identification of mentioned entities in conversational speech. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 953–956, New York, NY, USA. Association for Computing Machinery.

Tao Ge, Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. Automatic grammatical error correction for sequence-to-sequence text generation: An empirical study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6059–6064, Florence, Italy. Association for Computational Linguistics.

Sahar Ghannay, Antoine Caubrière, Yannick Estève, Nathalie Camelin, Edwin Simonnet, Antoine Laurent, and Emmanuel Morin. 2018. End-to-end named entity and semantic concept extraction from speech. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 692–699. IEEE.

Stephen Guo, Ming-Wei Chang, and Emre Kiciman. 2013. To link or not to link? a study on end-to-end tweet entity linking. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1020–1030, Atlanta, Georgia. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, Brussels, Belgium. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Thomas Lin, Mausam, and Oren Etzioni. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 84–88, Montréal, Canada. Association for Computational Linguistics.

Xiaohua Liu, Yitong Li, Haocheng Wu, Ming Zhou, Furu Wei, and Yi Lu. 2013. Entity linking for tweets. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1304–1311, Sofia, Bulgaria. Association for Computational Linguistics.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888.

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2019. Joint learning of named entity recognition and entity linking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 190–196, Florence, Italy. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Allen Schmaltz, Yoon Kim, Alexander Rush, and Stuart Shieber. 2017. Adapting sequence models for sentence correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2807–2813, Copenhagen, Denmark. Association for Computational Linguistics.

Avirup Sil and Alexander Yates. 2013. Re-ranking for joint named-entity recognition and linking. In *Proceedings of the 22nd ACM international conference on Information and Knowledge Management*, pages 2369—2374, San Francisco, USA.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.