Enhancing gazetteers for named entity recognition in conversational recommender systems

Nicholas Dingwall[1], Vianne R. Gao[2]

[1]Amazon, 525 Market St, San Francisco, CA 94105

[2]Weill Cornell Medicine, 1300 York Ave, New York, NY 10065 (Work conducted during an internship at Amazon)

Abstract

Named Entity Recognition (NER) is a crucial building block of a conversational agent, but remains challenging in real-word settings. It is particularly challenging for domains where the entities are linguistically complex and resemble common phrases (e.g. music and movies). While gazetteer features have been shown to improve NER performance, their utility is undermined by pervasive spurious entity matching. We propose a framework for gazetteer knowledge integration that incorporates external knowledge about entity popularity (e.g. a song's play count) to reduce spurious entity matching and improve the robustness of gazetteer features. Our experimental evaluations show that using unfiltered gazetteers degrades performance, but that incorporating external information improves it compared to a baseline model that doesn't use gazetteer information. Further, our framework can efficiently adapt to new entities in gazetteers without additional training, which is crucial for rapidly growing domains like music.

Keywords

natural language understanding, named-entity recognition, gazetteer, conversational recommender, music

1. Introduction

Voice assistants (Siri, Alexa, Google Assistant) are becoming increasingly popular and music has emerged as a primary use case for them [1, 2]. Without a screen for browsing, conversational recommenders are an appealing avenue to help users navigate their favorite music.

But four factors make identifying mentions of these entities difficult in the music domain. First, there are a lot of songs and artists: thousands of artists release millions of songs each year, and a modern deep learning system must store their names in its weights. Second, song and artist names can often resemble ordinary parts of speech, and so the system must disambiguate genuine references to musical entities from spurious matches. Third, users misremember the titles of songs or use abbreviations to refer to artists, limiting the applicability of canonical data sources. And fourth, new songs are continually being released – some of which immediately achieve their peak popularity – which obliges the owners of a model to regularly retrain the model.

Conversational systems make NER even more challenging: while single-turn commands are often well-structured and include indicators that a sequence tagging system can utilize to estimate a prior likelihood that a token represents an entity ("Alexa *play* X", etc), conversational responses lack such affordances. The system must also distinguish system-directed speech from background conversation overheard while waiting for a response: "who let the dogs out" is likely to be a song request (Baha Men), but "who let the cats out" is more likely to be a frustrated parent chastising their children. Finally, errors made by users in recalling an entity name or by a voice recognition system, alongside nonstandard spelling of artist and song titles, frustrate attempts at simple string matching against canonical entities [3].

Gazetteers – flat lists of entity names – can provide a source of valid entity names. But incorporating them into modern NER models has proved difficult (see Section 2.2), and the music domain makes their application even more precarious: any song title gazetteer will include common phrases like "yes" (LMFAO), "something like that" (Tim McGraw), and "stop" (Spice Girls), resulting in frequent false positive matches.

Nevertheless, incorporating them into models is appealing since they could allow a production system to generalize beyond examples seen during training, and to decouple updates to entity lists from model training.

In this paper, we experiment with utterance data and music domain knowledge data. In the conversational music recommender setting, a user is prompted to specify genres, moods or artists and hears samples of playlists matching the criteria they have provided so far (e.g. including a specified artist). The conversation continues until a sample is accepted, the user requests to play a specific song or artist, or the user explicitly ends the conversation or stops responding. The natural language

3rd Edition of Knowledge-aware and Conversational Recommender Systems (KaRS) & 5th Edition of Recommendation in Complex Environments (ComplexRec) Joint Workshop @ RecSys 2021, September 27–1 October 2021, Amsterdam, Netherlands

nickding@amazon.com (N. Dingwall); vrg4001@med.cornell.edu (V. R. Gao)
0000-0003-0026-2740 (N. Dingwall); 0000-0001-8990-0897 (V. R. Gao)

interpretation component must therefore be able to recognize any song or artist name mentioned by the user in order to select matching playlists to recommend and to avoid recommending playlists that do not match the user's request.

This paper explores different methods to extract value from gazetteers enriched with popularity information about songs, artists and albums. In all cases, we add token-level features indicating the presence or absence of that token (or sequence of tokens) within a gazetteer. We vary the preprocessing applied to the gazetteers and show that neither full gazetteers nor gazetteers filtered to include only the most popular entities outperform a baseline gazetteer-free model. However, after a more careful filtering of entities, adding a gazetteer does help the model to robustly extract music entity names. In doing so, the model improves its ability to classify a user's overall intent.

2. Background and prior work

2.1. Named entity recognition

Named entity recognition (NER) is the task of associating each word in a sentence with a label indicating its type. In typical settings, the type may be a person, a location, or an organization. In our domain, we are interested in music entities: artist names, song titles and album names.

In practice we refer to *tokens* instead of *words*, allowing for rare words to be split into subwords to limit the vocabulary size necessary to cover the entire dataset. For example, '*ed sheeran*' is represented as the three tokens '*ed*', '*sheer*' and '*an*'. We hope to train a model that associates all three tokens with the *artist_name* tag.

2.2. Gazetteers for NER

Gazetteers were common in pre-neural NER architectures: indeed, Mikheev *et al* in 1999 was notable for doing it *without* gazetteers [4].

Their use has fallen out of fashion with the recent dominance of large pre-trained language models for NER, since these models can better leverage contextual information to detect entity mentions [5]. More recent work has demonstrated that gazetteers can still improve NER performance with neural architectures, especially where training data is limited [6, 7, 8, 9, 10, 11].

However, the improved performance of modern NER models exposes the noise in gazetteers: Magnolini *et al* showed that filtering rarely-occurring values from large gazetteers boosts performance more than using the unfiltered gazetteer [7]. But in the music domain, the noise comes principally from linguistic ambiguity: entity names can be homographs of non-entity words and phrases. Filtering based on corpus frequency would retain many of these homographs (phrases like ”yes”, “something like that” and ”stop”), which are particularly common in conversational responses, and exclude many genuine references to entities. Moreover, we wish to use gazetteers precisely because they will help generalize beyond the training data, especially for low-context inputs, like an artist name on its own.

These works also either did not use pre-trained language models [6, 7, 8, 11] or did not fine-tune the weights of the language models [10, 9]. Large pre-trained language models based on the transformer architecture [12] have achieved state-of-the-art results across a variety of natural language processing tasks [13] but successfully integrating gazetteers remains elusive.

In these prior works, the gazetteers used were all flat lists of entity names, and so the systems could only consider the surface form of each entity (i.e. any string matching the name of the entity, regardless of the intended referent of that string). Oramas *et al* introduces a framework to leverage the popularity of each associated entity to distinguish between ambiguous and non-ambiguous names [14]. For each entity, they compute a ratio between the rank of the entity's popularity and the rank of the number of occurrences of its surface form in their corpus.

$$r(e) = \frac{popularityRank(e)}{frequencyRank(e)} \tag{1}$$

Mentions of entities that occur more frequently in their corpus than would be expected based on their popularity rank (i.e. $r(e)$ is small) are likely to be spurious matches. They use this to automatically label a training set: some entities can be confidently labeled as songs or artists, some – like “Could You”, “Play Music” and “Xmas” – are ignored, and inputs containing potentially-confusing mentions like “Country Joe” and “Spanish House” are excluded entirely. However, this informs only the dataset generation; their model does not have access to the underlying popularities or the rank.

Meng *et al* propose a mixture of experts model for NER that directly models how much weight to give to features derived from the context (using a BERT encoder) and from gazetteers (using a BiLSTM built on gazetteer matches) for each token. This substantially improves performance on their datasets, but still relies exclusively on linguistic features [15].

In this work, we show that filtering gazetteers using a similar formula to Equation 1 [14] allows an NER model to leverage gazetteer information. We compare against two other preprocessing methods which both degrade performance.

3. Datasets

3.1. Corpus

We train our model on historical user utterances. The data are labeled using a hand-crafted set of grammatical rules designed to match the most frequently-occurring utterances. The rules consist of a pattern and allowed slot values. For example, the pattern `just play <artist-name>` includes an `<artist-name>` slot; this slot is associated with a list of popular artists. The combination of pattern and slot list allows us to match “just play the beatles” and “just play rihanna”, but not “just play trivial pursuit”, since *trivial pursuit* is a game and not an artist.

In practice, we abstract common phrases and nest rules: `<negative-trigger> <artist-name>` matches a predetermined set of negative trigger phrases like “not”, “i don’t like”, etc, along with an artist name.

To control the latency of these rules, we limit slot lists to popular entities. The rules therefore fail on a long tail of infrequent utterances, either because the utterance contains an entity outside our canonical lists, or because the pattern is unusual. We observe that statistical models trained on these labeled utterances can generalize to long tail utterances, as proposed in [16].

The rules cover multiple intents, including standard ones like `YesIntent`, `NoIntent` and `StopIntent`, but also an `AddMusicConstraintIntent` for when users constrain recommended music entities. Slot types include entity types as well as trigger phrases that indicate negation, instructions to go immediately to playback, and so on.

To evaluate the model’s ability to discriminate between spurious mentions of entity names, we hand label independently-collected validation and test sets where each utterance contains a substring included in the song title gazetteer (see Section 3.2). We ignore the 50 most frequently-occurring spurious matches (e.g. “play”, “just”, “yeah”, etc) so that approximately 50% of utterances express a `AddMusicConstraintIntent`, and about 50% of these `AddMusicConstraintIntent` utterances contain a true reference to a specific song. Only `AddMusicConstraintIntent` can include song titles. These titles include some that were misremembered or that contain voice recognition errors.

Our grammatical rules can interpret about 30% of utterances in the test set and we consider these to be *in domain*; the remaining 70% test the model’s generalization capability, either to new utterance patterns or to novel entities.

The training, validation and test datasets are fixed for all experiments.

Table 1
Example showing which gazetteer embeddings trigger for each token in the utterance *play dancing queen.* In this case, *dancing queen* is recognized as a song, and *queen* as an artist. For simplicity, we assume no other gazetteer entries match the utterance.

entity type	play	dancing	queen
artist name	✗	✗	✓
song title	✗	✓	✓
album name	✗	✗	✗

3.2. Gazetteers

We use gazetteers derived from historical single-turn utterances that expressed a music request to a voice assistant. Since these utterances are well-structured (usually of the form “play X”), the entity name can be extracted and associated with an entity type by an entity resolution system.

These gazetteers can include entities with user or voice recognition errors as long as the entity resolution system was able to resolve them to a canonical entity.

As such, the gazetteers consist of multiple strings corresponding to the same entity: e.g. “blink one eighty two” and “blink one eight two” both appear in our gazetteer, even though the canonical name is “Blink 182”.

Each entity in a gazetteer is augmented with the number of times it was requested (which we refer to as its *popularity*).

4. Methodology

4.1. Candidate entity matching

Before passing a user’s utterance to the model, we must first determine which gazetteer entries appear in it. Note that at this stage, we do not distinguish between true positives (the user was referring to the entity) and false positives (spurious matches like ‘yes’, ‘play’, etc).

We find these candidates using a regex search for each gazetteer entry, enforcing that the match must terminate at whitespace or at the beginning or end of the utterance.

Next, we associate all of the tokens in an utterance with the entity type of the candidate. We summarize this information with a binary vector for each word, where each dimension corresponds to one of the gazetteers (artist name, song title, album name). See Table 1 for an example.

We add a dimension to this vector with its value fixed to 1 as a bias term. This entry can be thought of as the *no entity* dimension, which captures the possibility that the gazetteer matches were false positives.

4.2. Gazetteer filtering

In this work, we use a straightforward technique to incorporate the summary gazetteer vectors into our baseline model (see Section 4.3). The baseline model does not have access to this vector, and so we can evaluate whether gazetteers improve or degrade performance.

We compare three methods to preprocess the gazetteers:

First, we use the full, unfiltered gazetteer. Following Magnolini *et al* [7], we expect the noise introduced by false positive matches to outweigh any information provided by the true positives.

Second, we filter out the least popular entities in the gazetteer by thresholding the popularity (see Section 3.2). This is equivalent to using a shorter collection window to gather candidates. We expect that this does little to exclude ambiguous entities.

Third, we threshold the ratio between the number of occurrences of each entity's surface form in our training corpus and its popularity, similar to Oramas *et al* [14]. While Oramas *et al* used ranks, we use raw counts to capture the assumption that the number of genuine mentions of an entity is proportional to the underlying popularity of that entity. We call our version $\hat{r}$ to avoid confusion:

$$\hat{r}(e) = \frac{\text{popularity}(e)}{\text{mention\_frequency}(e)} \tag{2}$$

This has the practical benefit of allowing new entities to be added without re-ranking.

We rejected a fourth candidate method of thresholding based on the corpus frequency since the resulting filtered gazetteers preferentially included exactly the entities we wished to exclude, like "yes", "play", "stop", etc, and excluded entities not mentioned in our corpus, limiting a model's ability to generalize beyond its training data.

Where we filter gazetteers, we treat the percentage to filter out as a hyperparameter (25%, 50% or 75%) and select the model that performed best on the validation set.

4.3. Model

To understand a user's utterance, we need to predict the user's intent (classification) and label any entities they mentioned (NER). We start with a standard BERT-base model [17], pretrained on `book_corpus_wiki_en_uncased`[1].

To represent information from the gazetteers, we start by randomly-initialize four 64-dimensional 'ingredient' embeddings corresponding to the four-dimensional gazetteer vector described in Section 4.1 (*no entity*, *artist name*, *song title*, *album name*). Each token in the utterance is represented as the average of the ingredient embeddings for the entity types matched which that token appears in a candidate. Note that every token receives the *no entity* embedding as a bias term. This is illustrated in Figure 1.

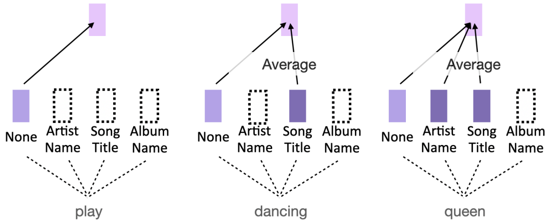

Figure 1: Gazetteer features are computed as the sum of associated 'ingredient' embeddings. Here, for example, 'queen' appears in the artist and song title gazetteers (via the artist "Queen" and the song "Dancing Queen"), so we take the average of those along with the *no entity* embedding.

We concatenate these gazetteer embeddings with the BERT output embeddings and add a single transformer encoder layer (i.e. self-attention with position embeddings and a fully-connected output layer) so that the gazetteer information can be shared among all the tokens.

The `[CLS]` token, which represents the entire utterance, receives the average embedding taken over all tokens in the utterance.

The outputs of the final transformer layer are passed to prediction heads for each token. The `[CLS]` token predicts the user intent, and the remaining tokens predict their own entity type label (or `OTHER`). Note that each token can have only one label, and the utterance is associated with exactly one intent.

This architecture is illustrated in Figure 2.

The baseline model is identical, except that nothing is concatenated with the BERT outputs.

The model is fine-tuned using cross-entropy with label smoothing [18], where the total loss is the sum of the classification loss and the slot tagging loss for each token. We update all parameters during fine-tuning, including the gazetteer 'ingredients'.

This architecture resembles the joint intent-classification and slot filling model introduced in Chen *et al* [19], except for the gazetteer embeddings, the additional transformer encoder layer, and the use of label smoothing. The first two of these additions provide a method to fuse gazetteer information into the model before the prediction heads. Label smoothing helps restrain the model's overconfidence on 'easy' examples, resulting in more robust performance on utterances outside the training distribution [18].

Aside from the percentage of each gazetteer to filter out

[1] From https://nlp.gluon.ai/model_zoo/bert/index.html

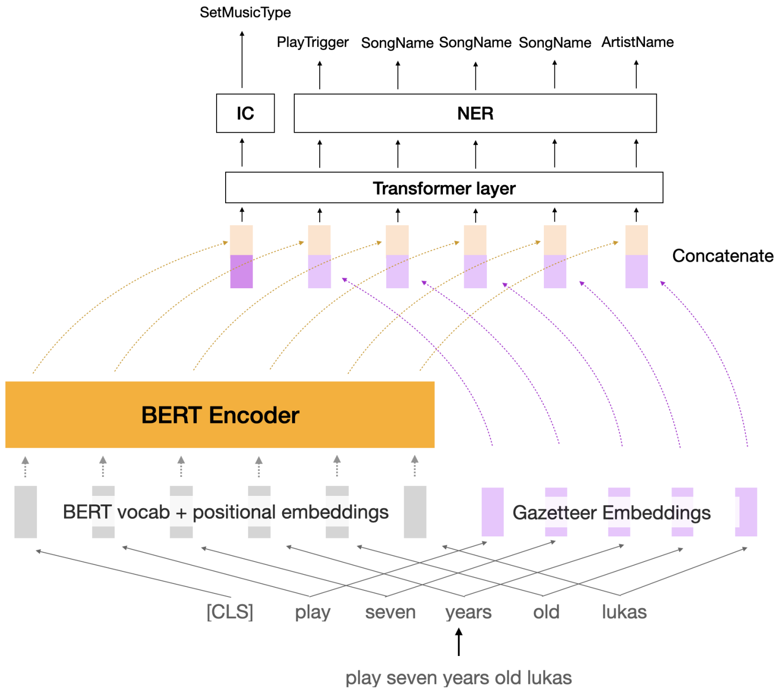

Figure 2: Model architecture. Contextual embeddings (from the BERT encoder) are concatenated with gazetteer embeddings (see Figure 1), and the resulting representation is passed through a transformer layer to prediction heads for both intent classification (IC) and entity labeling (NER).

(in the popularity-filtered and $\hat{r}(e)$-filtered experiment), we do not conduct any hyperparameter selection. We find in both cases that filtering out 75% of the gazetteer gives the best performance on the validation set. For other hyperparameters, we use values that previously performed well with a simplified baseline model that does not include the final transformer layer: since the utterances are typically short, we truncate them to 16 tokens (this affects fewer than 0.1% of utterances), use a batch size of 128, a label smoothing $\alpha = 0.1$, and train for 10,000 updates. We checkpoint every 100 updates and choose the version of the model that achieved the highest intent classification F1 score on the validation set. Other hyperparameters follow those in Chen *et al* [19].

5. Results

For each experiment, we evaluate the model's ability to discriminate `AddMusicConstraintIntent` from other intents, and its ability to extract correct song titles. Song title detection is particularly challenging for a conversational music recommender due to song titles' variability, cardinality and resemblance to normal speech.

We report this metric using the SemEval 'strict' methodology [20]. That means the span must exactly match the annotated span to be counted as a true positive; predicting the wrong span counts as both a false positive (the incorrectly-predicted span) and a false negative (the missed prediction). We choose this metric because we

Table 2
Results of experiments, shown as percentage increases or decreases from the baseline model.

(a) Song title detection.

Gazetteer	Precision	Recall	F1
None	-	-	-
Full	−2.72%	+0.58%	−1.17%
Popularity-filtered	−3.18%	+1.57%	−0.96%
$\hat{r}(e)$-filtered	+3.60%	+3.82%	+3.70%

(b) Intent classification (`AddMusicConstraintIntent`).

Gazetteer	Precision	Recall	F1
None	-	-	-
Full	+0.21%	−0.23%	−0.05%
Popularity-filtered	−0.21%	+1.85%	+0.98%
$\hat{r}(e)$-filtered	+0.25%	+2.58%	+1.70%

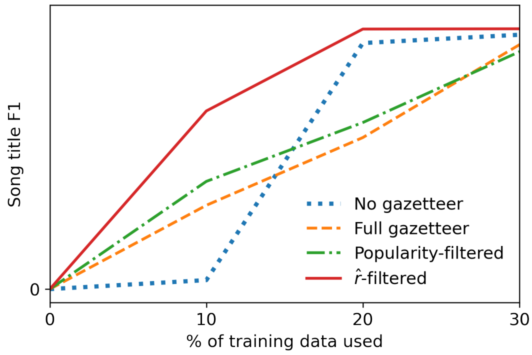

Figure 3: Song title F1 during training. Results shown every 10% up to 30% of training data, when F1 has begun to converge. Note that actual F1 scores are redacted due to their commercial sensitivity.

require substantially-complete predictions for the downstream entity resolver to associate the span with the correct entity. A simple token-by-token evaluation showed similar differences between models.

Table 2 shows the results of our experiments. As expected, we observe that using the full gazetteers increases the recall of song titles at the expense of precision, resulting in a drop in F1 score of 1.17%. Filtering based on popularity seems to exaggerate these differences, further diminishing precision but boosting recall even more, presumably because the model becomes too trusting of information from the gazetteers which still include spurious matches. The overall effect is that F1 dropped by slightly less: 0.96% from the baseline model.

Filtering based on the ratio $\hat{r}(e)$ addresses this issue. Common-but-spurious mentions are now excluded from the gazetteer, leaving a cleaner gazetteer that contains unambiguous entities, and which results in improved precision and recall and an overall increase in F1 of 3.70%.

These results seem to be correlated with intent classification performance, with the worst song title detection F1 corresponding to the worst intent classification F1 (full gazetteers), and best with best ($\hat{r}(e)$-filtered). This is to be expected: correctly recognizing the presence or absence of a song title (or artist name) makes distinguishing intents easier.

Figure 3 shows that the model with access to the $\hat{r}$-filtered gazetteer learns most quickly. The full and poularity-filtered gazetteers give an early boost to F1, when model performance is poor, but are quickly overtaken by the baseline model without gazetteers. This supports our hypothesis that information from noisy gazetteers helps weak models, but when the model is better able to leverage contextual cues, the noise begins to dominate any signal they provide. The model would by now perform better by ignoring the information, but it may have approached a local minimum in the loss surface from which it cannot escape, resulting in poorer performance at convergence (as shown in Table 2).

Table 3 shows some example user inputs that highlight how gazetteers help the model. Each utterance is shown with the song title predicted by the model learned under each experiment. While all the models are usually able to detect the presence of a song title, only the model trained using the $\hat{r}(e)$-filtered gazetteers is able to reliably detect the boundaries of the mention.

6. Limitations and future work

We note that this work only considers utterances in English. The technique described here should apply to other languages, but in some, whitespace cannot be used to delimit entities, making candidate matching more challenging.

We only briefly experimented with the impact of changes to the gazetteer after model training (e.g. due to new releases or changing popularity of existing releases). While these initial results are promising, we would want to conduct more thorough research to evaluate how predictions are affected.

We have also not explored the impact of false negatives (i.e. real entities not matched in the gazetteers, either because the entity is not sufficiently popular, because it has been recently released, or due to a user or voice recognition error). Our evaluation shows an overall improvement in precision and recall, but there may be individual cases where the baseline model better leverages contextual clues to predict entity mentions. Randomly dropping out gazetteer features during training (i.e. replacing a 1 with a 0 in the gazetteer vector described in Section 4.1 some fraction of the time) might force a model to learn how to use gazetteer features where available, but to continue to attend to contextual information otherwise, further improving overall performance.

This work was evaluated on manually-annotated offline datasets, but we have planned an A-B test to measure the downstream impact of improved NER performance on the rate with which users accept the system's recommendations. We expect to see an improvement corresponding to the system's ability to correctly interpret our users' wishes.

In future work, we intend to fuse popularity and $\hat{r}(e)$ directly into the model, rather than using it to filter the gazetteers. Incorporating the ratio $\hat{r}(e)$ into the model as a feature would allow it to attend more heavily to gazetteer features where the entity is unambiguous, and use contextual cues to disambiguate less obvious examples. It also avoids introducing an arbitrary cut off: small values of $\hat{r}(e)$ would be almost, but not quite, equivalent to excluding the entity entirely. We hope that such

Table 3
Examples of errors made by models trained with different gazetteer information. The expected song titles are underlined. Note that the model handles over a dozen intents, and so identifying song titles even in somewhat structured utterances (e.g. "X by Y") is nontrivial.

Utterance	No gazetteers	Full gazetteers
rolling in the deep	in the deep	in the deep
play cruella de vil		cruella de
high voltage		
just the way you are by	the way you are	the way you are
you dropped the bomb on me	dropped the bomb on me	dropped the bomb on me
green eyed lady by sugarloaf	eyed lady	eyed lady
monsters by shinedown		

Utterance	Popularity-filtered gazetteers	$\hat{r}(e)$-filtered gazetteers
rolling in the deep	rolling in the deep	rolling in the deep
play cruella de vil	cruella de vil	cruella de vil
high voltage		high voltage
just the way you are by		just the way you are
you dropped the bomb on me	dropped the bomb on me	you dropped the bomb on me
green eyed lady by sugarloaf	eyed lady	green eyed lady
monsters by shinedown		monsters

an approach will yield further improvements and be a step towards a general approach to integrating gazetteers with pre-trained transformers.

7. Conclusion

In this paper, we demonstrate that a rather simple architecture with carefully filtered gazetteers can greatly improve NER performance in a conversational recommendation system for the music domain. By augmenting gazetteers with information about the underlying likelihood of a mention of each entity, the models can avoid false positives, and are better able to rely on large gazetteers.

This finding could apply to other domains where large gazetteers are common and where relevant frequency information is available. Examples might include place names along with their populations, or diseases with the number of diagnoses mentioned in discharge notes.

Acknowledgments

The authors would like to thank Tao Ye, Justin Hugues-Nuger, Chelsea Weaver and Vlad Magdin for their help through conversations regarding the evaluation, technical implementation and presentation of this work. We also thank the reviewers for their valuable comments.

References

[1] T. Ammari, J. Kaye, J. Y. Tsai, F. Bentley, Music, Search, and IoT: How people (really) use voice assistants, ACM Transactions on Computer-Human Interaction 26 (2019). doi:10.1145/3311956.

[2] J. Thom, A. Nazarian, R. Brillman, H. Cramer, S. Mennicken, "Play Music": User Motivations and Expectations for Non-Specific Voice Queries, in: 21st International Society for Music Information Retrieval Conference, 2020.

[3] C. Gao, W. Lei, X. He, M. de Rijke, T.-S. Chua, Advances and challenges in conversational recommender systems: A survey, AI Open 2 (2021) 100–126. URL: https://doi.org/10.1016/j.aiopen.2021.06.002. doi:10.1016/j.aiopen.2021.06.002. arXiv:2101.09459.

[4] A. Mikheev, M. Moens, C. Grover, Named Entity recognition without gazetteers, in: Proceedings of EACL '99, 1999, p. 1. doi:10.3115/977035.977037.

[5] S. Peshterliev, C. Dupuy, I. Kiss, Self-Attention Gazetteer Embeddings for Named-Entity Recognition (2020). URL: http://arxiv.org/abs/2004.04060. arXiv:2004.04060.

[6] S. Rijhwani, S. Zhou, G. Neubig, J. Carbonell, Soft Gazetteers for Low-Resource Named Entity Recognition, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 8118–8123. doi:10.18653/v1/2020.acl-main.722. arXiv:2005.01866.

[7] S. Magnolini, V. Piccioni, V. Balaraman, M. Guerini, B. Magnini, How to Use Gazetteers for En-

tity Recognition with Neural Models, in: Proceedings of the 5th Workshop on Semantic Deep Learning (SemDeep-5), 2019, pp. 40–49. URL: https://github.com/XuezheMax/NeuroNLP2%0Ahttps://www.aclweb.org/anthology/W19-5807.

[8] T. Liu, J. G. Yao, C. Y. Lin, Towards improving neural named entity recognition with gazetteers, in: Proceedings ofthe 57th Annual Meeting ofthe Association for Computational Linguistics, 2019, pp. 5301–5307. doi:10.18653/v1/p19-1524.

[9] C. H. Song, D. Lawrie, T. Finin, J. Mayfield, Improving Neural Named Entity Recognition with Gazetteers, in: The 33rd International FLAIRS Conference, 2020, p. 8. URL: https://arxiv.org/abs/2003.03072.

[10] H. Lin, Y. Lu, X. Han, L. Sun, B. Dong, S. Jiang, Gazetteer-Enhanced Attentive Neural Networks for Named Entity Recognition, in: Proceedings ofthe 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, 2019, pp. 6232–6237.

[11] O. Agarwal, A. Nenkova, The Utility and Interplay of Gazetteers and Entity Segmentation for Named Entity Recognition in English, in: Findings ofthe Association for Computational Linguistics: ACL-IJCNLP, Association for Computational Linguistics, 2021, pp. 3990–4002. doi:10.18653/v1/2021.findings-acl.349.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[13] A. Chernyavskiy, D. Ilvovsky, P. Nakov, Transformers: "The End of History" for NLP? (2021). URL: http://arxiv.org/abs/2105.00813. arXiv:2105.00813.

[14] S. Oramas, M. Quadrana, F. Gouyon, P. M. Llc, Bootstrapping a Music Voice Assistant with Weak Supervision, in: Proceedings of NAACL HLT 2021: Industry Track, 2021, pp. 49–55.

[15] T. Meng, A. Fang, O. Rokhlenko, S. Malmasi, GEMNET: Effective Gated Gazetteer Representations for Recognizing Complex Entities in Low-context Input, in: Proceedings of the 2021 Conference of the North American Chapter of theAssociation for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, 2021, pp. 1499–1512. doi:10.18653/v1/2021.naacl-main.118.

[16] M. Mintz, S. Bills, R. Snow, D. Jurafsky, Distant supervision for relation extraction without labeled data, in: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2009, pp. 1003–1011. URL: https://aclanthology.org/P09-1113. doi:10.3115/1690219.1690287.

[17] J. Devlin, M. W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference 1 (2019) 4171–4186. arXiv:1810.04805.

[18] R. Müller, S. Kornblith, G. Hinton, When does label smoothing help?, in: Advances in Neural Information Processing Systems, 2019. arXiv:1906.02629.

[19] Q. Chen, Z. Zhuo, W. Wang, Bert for joint intent classification and slot filling, 2019. arXiv:1902.10909.

[20] D. S. Batista, Named-entity evaluation metrics based on entity-level, 2019. URL: http://www.davidsbatista.net/.