GEMNET: Effective Gated Gazetteer Representations for Recognizing Complex Entities in Low-context Input

Tao Meng* UCLA Los Angeles, USA

Anjie Fang Amazon.com, Inc. Seattle, WA, USA

Oleg Rokhlenko Amazon.com, Inc. Seattle, WA, USA

Shervin Malmasi Amazon.com, Inc. Seattle, WA, USA

tmeng@cs.ucla.edu {njfn,olegro,malmasi}@amazon.com

Abstract

Named Entity Recognition (NER) remains difficult in real-world settings; current challenges include short texts (low context), emerging entities, and complex entities (e.g. movie names). Gazetteer features can help, but results have been mixed due to challenges with adding extra features, and a lack of realistic evaluation data. It has been shown that including gazetteer features can cause models to overuse or underuse them, leading to poor generalization. We propose GEMNET, a novel approach for gazetteer knowledge integration, including (1) a flexible Contextual Gazetteer Representation (CGR) encoder that can be fused with any word-level model; and (2) a Mixture-of-Experts gating network that overcomes the feature overuse issue by learning to conditionally combine the context and gazetteer features, instead of assigning them fixed weights. To comprehensively evaluate our approaches, we create 3 large NER datasets (24M tokens) reflecting current challenges. In an uncased setting, our methods show large gains (up to $+49\%$ F1) in recognizing difficult entities compared to existing baselines. On standard benchmarks, we achieve a new uncased SOTA on CoNLL03 and WNUT17.

1 Introduction

Identifying entities is a core NLP task. Named Entity Recognition (NER) is the task of finding entities and recognizing their type (e.g., person or location). Mention Detection (MD) is a simpler task of identifying entity spans, without the types.

Advances in neural NER have produced high scores on benchmark datasets like CoNLL03 and OntoNotes (Devlin et al., 2019). However, a number of challenges remain. As noted by Augenstein et al. (2017), these scores are driven by the use of well-formed news text, the presence of "easy" entities, and memorization effects due to entity overlap between train/test sets; these models perform significantly worse on unseen entities or noisy text.

*This research was done during an internship at Amazon.

1.1 Current NER Challenges

Beyond news text, many challenges remain in NER. Context information has been shown to be important for NER (Jayarao et al., 2018), and short texts like search queries are very challenging due to low context and a lack of surface features (Guo et al., 2009; Carmel et al., 2014). Unseen and emerging entities also pose a challenge (Bernier-Colborne and Langlais, 2020). Finally, some entities, like movie names are not simple noun phrases and are harder to recognize (Ashwini and Choi, 2014). Table 1 lists more details about these challenges, and how they can be evaluated.

Entity Knowledge is essential for overcoming these issues, and critical in the absence of casing. Even a human may not correctly parse "`what is [[life is beautiful]]?`" without knowing that a movie is being referenced. However, most models start with no knowledge of real world entities, learning them from the training data. Continuous data annotation can add new entities, but is expensive and often not feasible.

Consequently, methods for integrating external knowledge, e.g., Knowledge Bases (KBs) or gazetteers, into neural architectures have gained renewed attention. However, such studies have reported limited gains (Liu et al., 2019; Rijhwani et al., 2020). The mixed success of gazetteers stems from three main limitations in current work: gazetteer feature representation, their integration with contextual models, and a lack of data.

For the representation, one-hot binary encoding is often used to represent gazetteer features (Song et al., 2020). However, this does not capture contextual info or span boundaries. Alternatively, independent span taggers trained on gazetteers have been proposed to extract potential entities Liu et al. (2019), but such models can be difficult to train and may not provide reliable features.

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1499–1512
June 6–11, 2021.

Challenge	Description
Short Texts For: voice, search	News texts have long sentences discussing many entities, but other use cases (search queries, questions) have shorter inputs. Datasets with minimal context are needed to assess performance of such use cases. Capitalization/punctuation features are large drivers of success in NER (Mayhew et al., 2019), but short inputs (ASR, user input) often lack these surface feature. An uncased evaluation setting is needed to understand model performance.
Long-tail Entities For: domains with many entities	In many domains entities have a large long-tail distribution, with millions of values (e.g., location names). This makes it hard to build representative training data, as it can only cover a portion of the potentially infinite entity space. A very large test set is required for effective evaluation.
Emerging Entities For: domains with growing entities	All entity types are open classes (new ones are added), but some groups have a faster growth rate, e.g., new books/songs/movies are released weekly. Assessing true generalization requires test sets with many unseen entities, to mimic an open-world setting.
Complex Entities For: voice, search	Not all entities are proper names: some types (e.g. creative works) can be linguistically complex. They can be complex noun phrases (`Eternal Sunshine of the Spotless Mind`), gerunds (`Saving Private Ryan`), infinitives (`To Kill a Mockingbird`), or full clauses (`Mr.Smith Goes to Washington`). Syntactic parsing of such nouns is hard, and most current parsers/NER systems fail to recognize them. The top system from WNUT 2017 achieved 8% recall for creative work entities (Aguilar et al., 2017). Effective evaluation requires corpora with many such entities.

Table 1: NER challenges not addressed by current work and datasets, and proposed solutions.

There are also limitations in the integration of gazetteer features. Existing studies often add extra features to a word-level model's Contextual Word Representations (CWRs), which typically contain no info about real world entities or their spans (Yamada et al., 2020). This concatenation approach is sub-optimal as it creates additional, and often highly correlated features. This has been shown to cause feature "under-training", where the model will learn to mostly rely on either context or gazetteer during training, and underuse the other (Yang et al., 2016). This can be problematic as the utility of the gazetteer is variable: it is valuable in low-context cases, but may not be useful when rich syntactic context (from the CWR) can identify entities. Conversely, a true entity may be missing from the gazetteer. However, when gazetteers are represented as an independent feature, the model assigns it a fixed weight, and its contribution to the prediction is static. To overcome this, external knowledge should dynamically be infused into relevant dimensions of the CWR, with the model learning to conditionally balance the contribution of the CWR and gazetteer to the prediction.

Finally, these issues are compounded by a lack of data reflecting the challenges from Table 1, which prevents the exploration of effective architectures for knowledge injection.

1.2 Our Contributions

The key contributions of this paper are new data and methods to address the above challenges.

We propose GEMNET, a gazetteer expert mixture network for effectively integrating gazetteers into any word-level model. The model includes an encoder for **C**ontextual **G**azetteer **R**epresentations (CGRs) as a way to incorporate any number of gazetteers into a single, span-aware, dense representation. We also propose a gated Mixture-of-Experts (MoE) method to fuse CGRs with Contextual Word Representations from any word-level model (e.g., BERT), something not explored in previous work. Our novel MoE approach allows the model to conditionally compute a joint CGR-CWR representation, training a gating network to learn how to balance the contribution of context and gazetteer features. Finally, we employ multi-stage training to drive further improvements by aligning the CGR/CWR vectors.

To evaluate our proposed approaches, we create 3 challenging NER datasets that represent short sentences, questions, and search queries. The created datasets have complex entities with low-context and represent the challenges in Table 1.

Extensive experiments in an uncased setting show that our MoE model outperforms other baselines, including concatenation, in all experiments. We achieve state-of-the-art (SOTA) results on CoNLL03/WNUT17, but its utility is more notable on our difficult low-context data. We show that short texts make NER much harder, but gazetteers yield huge gains of up to $+49\%$ F1, specially in recognizing complex/unseen entities. We also show that gazetteer coverage during training is important.

2 Related Work

Deep Learning for NER Neural approaches have greatly improved NER results in recent years. A shift to encoders e.g., BiLSTM-CRF models (Huang et al., 2015), using static word embed-

dings eliminated the need for manual feature engineering (e.g., capitalization features). More recently, transformer-based Language Models (LMs), e.g., BERT (Devlin et al., 2019), achieved further improvements by using deep contextual word representations. Such models jointly learn syntactic cues and entity knowledge, and may fail to recognize unseen or syntactically ambiguous entities. Consequently, training data is augmented with gazetteers.

NER with Gazetteers Annotated NER data can only achieve coverage for a finite set of entities, but models face a potentially infinite entity space in the real world. To address this, researchers have integrated gazetteers into models (Bender et al., 2003; Malmasi and Dras, 2015). String matching is commonly used to extract gazetteer matches, which are then concatenated to word representations. Song et al. (2020) use gazetteers from the Wikidata KB to generate one-hot vectors that are concatenated to BERT representations, yielding minor improvements on CoNLL03. This concatenation approach has been shown to cause feature "under-training" (Yang et al., 2016), as discussed in §1. An alternative approach uses gazetteers to train a subtagger model to recognize entity spans. Liu et al. (2019) propose a hybrid semi-Markov CRF subtagger, reporting minor improvements. While a subtagger may learn regularities in entity names, a key limitation is that it needs retraining and evaluation on gazetteer updates. Recent work has considered directly integrating knowledge into transformers, e.g., KnowBert adds knowledge to BERT layers (Peters et al., 2019), and LUKE is pretrained to predict masked entities (Yamada et al., 2020). The drawbacks of such methods are that they are specific to Transformers, and the model's knowledge cannot be updated without retraining. We aim to overcome the limitations of previous work by designing a model-agnostic gazetteer representation that can be fused into any word-level model.

Mixture-of-Experts (MoE) Models MoE is an approach for conditionally computing a representation, given several expert inputs, which can be neural models with different architectures (Arnaud et al., 2020) or models using different knowledge sources (Jain et al., 2019). In MoE, a gating network is trained to dynamically weight experts per-instance, according to the input. It has demonstrated to be useful in various applications like recommendation (Zhu et al., 2020), domain adaptation for sentiment analysis, and POS tagging (Guo et al., 2018). For NER, Liu et al. (2020) proposed a Mixture of Entity Experts (MoEE) approach where they train an expert layer for each entity type, and then combine them using an MoE approach. Their approach does not include external gazetteers, and the experts provide an independent representation that is not combined with the word representation. In our work we treat word and external gazetteer representations as independent experts, applying MoE to learn a dynamically fused representation.

3 Datasets

We experiment using three standard benchmarks: CoNLL03, OntoNotes, and WNUT17. However, these corpora do not capture the issues from Table 1; rich context and common entities (country names) allow a simple RNN model to achieve near-SOTA results. A key contribution of our paper is the creation of 3 new datasets that represent those challenges. They are difficult, as shown in §5.1.

NER Taxonomy: We adopt the WNUT 2017 (Derczynski et al., 2017) taxonomy entity types: `PERSON` (`PER` for short, names of people), `LOCATION` (`LOC`, locations/physical facilities), `CORPORATION` (`CORP`, corporations and businesses), `GROUPS` (`GRP`, all other groups), `PRODUCT` (`PROD`, consumer products), and `CREATIVE-WORK` (`CW`, movie/song/book/etc. titles).

Our datasets are described below.[1] All data are uncased, and we make them publicly available.[2] Their statistics, listed in Table 2, show that they reflect the challenges from §1: short inputs (low context), with many unseen entities in the test set.

LOWNER (Low-Context Wikipedia NER) To create our training set, we take advantage of the rich interlinks in Wikipedia. We parse the English Wikpedia dump and extract sentences from all articles. The sentences are parsed, and linked pages are resolved to their respective Wikidata entities to identify their type. To mimic search and voice settings, we minimize the context around the entities by dropping sentences with unlinked entities, identified using interlinks and a capitalization heuristic. The result is a corpus of 1.4 million low-context sentences with annotated entities, e.g., "`A version for the [sega cd] was also announced.`"

[1] More details about their development are in Appendix A

[2] https://registry.opendata.aws/lowcontext-ner-gaz

Set	Dataset	Type	# Sentence	# Token	# Entity	Avg. Sent Len	Entity Type Distribution					
							PER	LOC	CORP	GRP	PROD	CW
1	LOWNER	Train	13,424	206,772	13,555	15.40±6.35	5,029	3,791	631	1,941	424	1,805
2	LOWNER	Dev	3,366	51,651	3,813	15.34±6.28	1,255	1,235	169	565	101	499
3	LOWNER	Test	1,385,290	21,303,399	490,749	15.37±6.29	215,411	120,480	20,015	52,566	15,976	74,830
4	MSQ-NER	Test	17,868	98,117	18,993	5.49±1.86	4,586	10,468	679	610	469	2,187
5	ORCAS-NER	Test	471,746	1,958,020	368,250	4.15±1.75	68,000	162,652	28,738	23,058	18,114	71,461

Table 2: Data statistics. Entity counts are unique values. LOWNER has train/dev/test sets, the rest are test sets.

MSQ-NER: MS-MARCO Question NER To represent NER in the QA domain, we create a set of natural language questions, based on the MS-MARCO QnA corpus (V2.1) (Bajaj et al., 2016). Like Wu et al. (2020), we templatize the questions by applying NER to extract item names, which are then mapped to our taxonomy. Entities are replaced with their types to create templates, e.g., "`who sang <CW>`" and "`when did <PROD> come out`". Approx 3.5k Templates (appearing $>= 5$ times) are chosen and slotted with entities from a knowledge base to generate 18k annotated questions e.g., "`when did [xbox 360] come out`". There are a wide range of question shapes and entity types, please see Appendix A for examples.

ORCAS-NER: Search Query NER To represent the search query domain, we utilize 10 million Bing user queries from the ORCAS dataset (Craswell et al., 2020) and apply the same templatization procedure as MSQ-NER. This yields search templates e.g., "`<PROD> price`" and "`<CORP> phone number`", which are used to create annotated queries, e.g., "`[airpods pro] reviews`". A total of 472k queries are generated from 97k unique templates, please see examples in Appendix A.

3.1 Gazetteer Data

Our gazetteer is composed of 1.67 million entities from the English Wikidata KB. Instead of collecting entities from the web (Khashabi et al., 2018), we focused on entities that map to our taxonomy. Alternative names (aliases) for entities are included. Gazetteer statistics are listed in Appendix B.

4 The GEMNET Model

We propose GEMNET, a generic gazetteer fusion approach that can be integrated with any word-level model, e.g., RNNs and Transformers. We experiment with both BiLSTM-CRF and BERT-CRF models which produce (contextual) word representations, and complement these "word experts" with gazetteers. The overall architecture is shown in Figure 1, and the components are detailed below.

4.1 Contextual Gazetteer Representations

Our gazetteer representations is obtained in two steps: entry matching, and contextual encoding.

	O	B-PROD	I-PROD	B-CORP	I-CORP
How	1	0	0	0	0
much	1	0	0	0	0
is	1	0	0	0	0
Apple	0	1	0	1	0
iPhone	0	1	1	0	0
12	0	0	1	0	0

Table 3: Example of our gazetteer representation.

Gazetteer Entry Matching A gazetteer $\mathbf{g}$ is a list of entries that are associated with a category. For instance, a `PERSON` gazetteer contains a list of known people. The k-th entry $\mathbf{g}^{(k)}$ is associated with a tokenized string (`'John Carpenter'`) and $\mathbf{t}^{(k)}$ holds the IOB2 tags (`[B-PER, I-PER]`). We use T to denote the tag set over all gazetteers, e.g., $T = \{$`B-PER, I-PER, B-LOC, I-LOC, O, ...`$\}$.

We denote input sentences as $(w_1, w_2, \ldots, w_L)$, where w_i is the i-th word, and L is the length. Full string matching is applied to inputs to identify matches across all gazetteers. Overlapping matches are resolved by preferring longer ones over shorter ones, and earlier matches over later ones. A match matrix, $\mathbf{M} \in \{0, 1\}^{L \times |T|}$, represents the matching results. It is initialized with zeros, and successful matches $(w_i, w_{i+1}, \ldots, w_{i+m}) = \mathbf{g}^{(k)}$ will set

$$\mathbf{M}_{i+j, \mathbf{t}_j^{(k)}} = 1,\ j = 0, 1, \ldots, m,$$

indicating that the word w_{i+j} is represented by a one-hot vector over the tag set T.

A key advantage of this representation is that it captures multiple matches for a given span in a sentence. As shown in Table 3, the word "apple" can be matched to product and organization types. Furthermore, it is span-aware due to the IOB2 encoding. Any number of types and gazetteers can be added as needed, allowing the model to learn from correlations, and identify ambiguous entities.

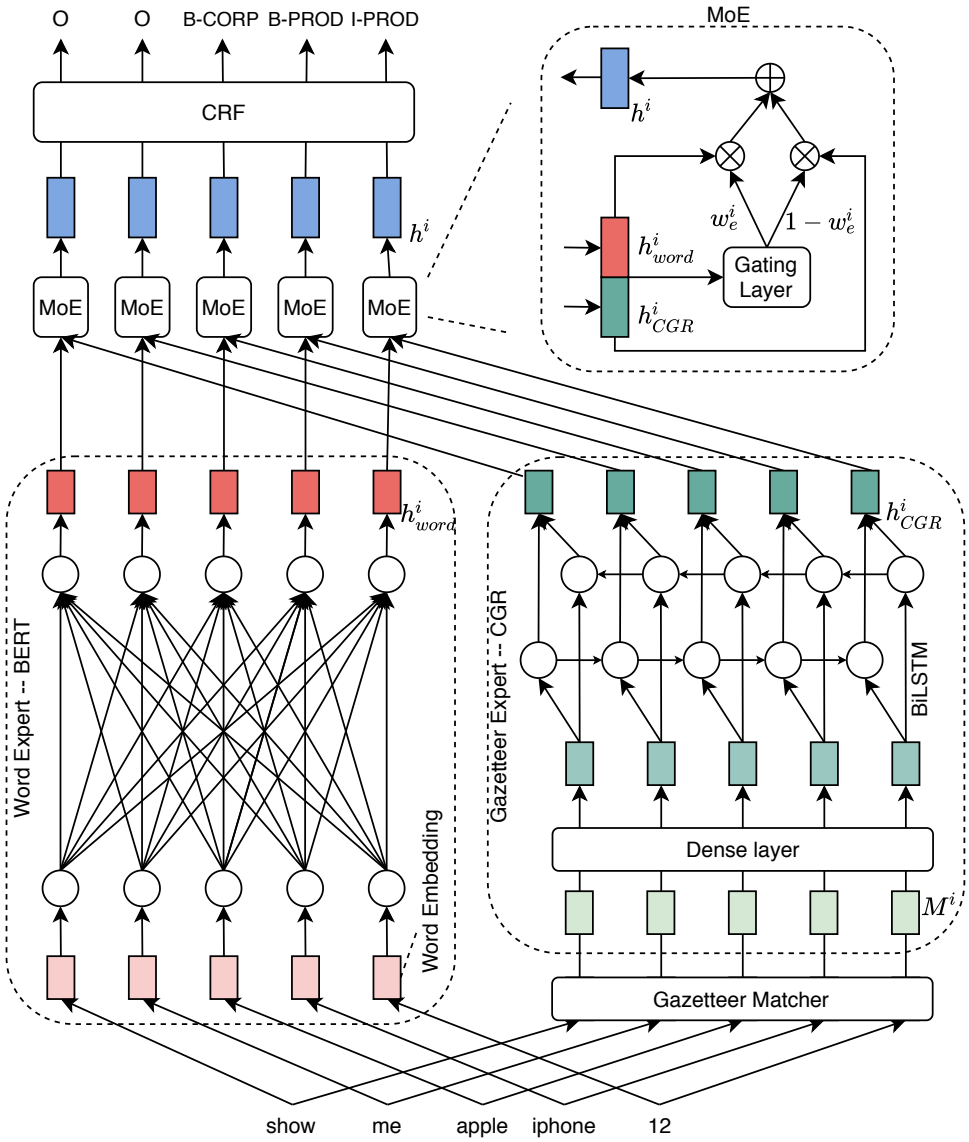

Figure 1: GEMNET model architecture. Input is passed to the word expert (e.g. BERT) and its match matrix to the Contextual Gazetteer Representation (CGR) encoder. Their outputs are dynamically combined via a Mixture-of-Experts (MoE) gating network and passed to a CRF layer for word prediction.

$\mathbf{M}$ is extracted by a gazetteer matcher, as a preprocessing step outside the network. This modular approach has an important advantage: it allows the gazetteer to be updated without model retraining. This is useful for efficiently recognizing emerging entities, and supporting personalized user-defined entities (e.g., contact lists).

Contextual Encoding $\mathbf{M}$ can be directly used as input features, but is sparse. We use a linear projection to obtain a dense representation per word:

$$\mathbf{h}^i_{gaz} = f(\mathbf{w} \cdot \mathbf{M}_i + \mathbf{b})$$

where $\mathbf{w} \in \mathbb{R}^{D \times T}$ and $\mathbf{b} \in \mathbb{R}^D$ are trainable parameters, D is the hidden dimension of gazetteer representation and f is an activation function. This creates a dense representation that captures interactions between multiple matches. We then contextualize this representation by applying a BiLSTM:

$$\mathbf{h}^i_{forward} = \text{LSTM}(\mathbf{h}^{i-1}_{forward}, \mathbf{h}^i_{gaz})$$
$$\mathbf{h}^i_{backward} = \text{LSTM}(\mathbf{h}^{i+1}_{backward}, \mathbf{h}^i_{gaz})$$
$$\mathbf{h}^i_{CGR} = [\mathbf{h}^i_{forward}, \mathbf{h}^i_{backward}]$$

where $[\cdot, \cdot]$ is the concatenation. A sample visualization of the embeddings is shown in Appendix D.

This dense **contextualized gazetteer representation** (CGR) can capture information about entity span boundaries (present in $\mathbf{M}$), as well as interactions between entities in a sentence.

4.2 Gazetteer Knowledge (CGR) Integration

The CGR operates on IOB2 tags and cannot memorize specific patterns; it is designed to be integrated with a lexical model. We consider these representations to be orthogonal: CGRs can complement the model's knowledge and syntactic representation.

CGR Concatenation The simplest integration is to concatenate the dense CGR to the CWR, while jointly training the two representations.

Mixture-of-Experts (MoE) Model The word-level model and CGRs complement each other and may not always be in agreement. The word model may have low confidence about the span of an unseen entity, but the gazetteer may have knowledge of it. Conversely, the model's syntactic context may be confident about a span not in the gazetteer.

In fact, the two sources can be considered as independent experts and an effective model should learn to use their outputs dynamically. Inspired by the MoE architecture (Pavlitskaya et al., 2020), we apply conditional computation to combine our representations, allowing the model to learn the contexts where it should rely more on each expert.

We add a gating network to create a weighted linear combination of the word and gazetteer representations. For a sentence, the two models output[3] their representations $\mathbf{h}_{word}$ and $\mathbf{h}_{gaz}$, which are used to train the gating network:

$$w_e = \sigma(\theta[\mathbf{h}_{word}, \mathbf{h}_{CGR}]),$$
$$\mathbf{h} = w_e \cdot \mathbf{h}_{word} + (1 - w_e) \cdot \mathbf{h}_{CGR},$$

where θ are trainable parameters with size $2L$, $[\cdot, \cdot]$ is the concatenation and σ is the Sigmoid activation function. We learn gating weights, w_e, so that the model can learn to dynamically compute the hidden information $\mathbf{h}$ for each word. The architecture of our model is shown in Figure 1. After obtaining $\mathbf{h}$, we feed it to a CRF layer to predict a tag.

Two-stage Training Our architecture jointly optimizes over both experts, but their initial states differ. The word expert often contains pretrained elements, either as word embeddings or transformers. The randomly-initialized CGR will have high initial loss, and its representation is not aligned with the word expert, preventing correct convergence. We tackle this problem through a two-stage training method to adapt the two experts to each other. In the first stage, we freeze the word expert and only train the CGR encoder with the MoE and CRF layers, forcing the model to use gazetteer knowledge in order to minimize the loss. Importantly, this also adapts the CGR encoder to align its representation with that of the word expert, e.g., the dimensions with noun signals will be aligned with those of BERT, enabling the computation of their linear combination. In the second stage, the two experts are jointly fine-tuned to co-adapt them. This ensures that the CGR encoder starts with reasonable weights, and allows the MoE gating network to better learn how to balance the two experts.

[3] Outputs sizes must be equal, e.g., CGR must match BERT.

5 Experiments

Data: All experiments are uncased, using standard benchmarks (CoNLL03, OntoNotes, WNUT17) and the new datasets we create (see §3).

Models: We integrate GEMNET with both BERT and BiLSTM word encoders.For BERT, we use the pretrained $\text{BERT}_{\text{BASE}}$ model. The last output layer is used, and for each word, we use the first wordpiece representation as its representation. The BiLSTM model has 3 inputs: GloVe embeddings (Pennington et al., 2014), ELMo embeddings (Peters et al., 2018) and CharCNN embeddings (Ma and Hovy, 2016).

Evaluation: We evaluate MD and NER, and report entity-level precision, recall and F1 scores.

5.1 MD Baselines

Our first experiment aims to measure the difficulty of our datasets (§3) relative to existing benchmarks. We train a BERT model on CoNLL03 and use it to measure MD performance on our data. Measuring NER performance is not possible as we use a different tag set (WNUT17 vs CoNLL03).

Dataset	P	R	F1
CoNLL03	96.9	95.7	96.3
LOWNER	67.5	74.5	70.9
MSQ-NER	38.9	38.7	38.8
ORCAS-NER	56.8	51.6	54.1

Table 4: Mention detection (MD) results for a BERT model trained on CoNLL03, tested on our data.

Results: Compared to the CoNLL03 results, the LOWNER performance is worse. Although the evaluation on LOWNER is a transfer setting, the large gap shows the existing model cannot generalize

well to our datasets due to the hard entities. Results for MSQ-NER and ORCAS-NER, which are short texts, are even lower. Overall, we note the difficulty of our datasets due to low context and hard entities.

5.2 NER Ablation Experiments

We explore all model architectures by training on LOWNER (set 1 in Table 2) and evaluating MD and NER performance on all datasets (sets 3–5 in Table 2). See Appendix C for training details.

Models: The GEMNET model is jointly trained and fused with BERT and BiLSTM word encoders, with and without two-stage training. To assess the impact of the MoE component, we also concatenate the CGR and CWR vectors, without MoE.

Baselines: We compare against three baselines: (1) no gazetteer baselines; (2) binary concatenation: we simply concatenate the binary match features (M) to the word representations, as is common in the literature; (3) the subtagger model of Liu et al. (2019). They are shown as "baselines" in table 5.

Results: MD and NER performance for all models is shown in Table 5. Overall we note the high effectiveness of the GEMNET model. In particular, our BiLSTM-based GEMNET approach improves F1 by up to 49% over the no gazetteer BiLSTM baseline in ORCAS-NER. Different aspects of the results are discussed below.

Word Encoder Performance: For LOWNER, we note that BERT achieves the best results, which is to be expected since the data consists of full sentences. MD is easier than NER, and represents the upper bound for NER. Performance in all cases decreases with low context, with search queries (ORCAS-NER) being the hardest. BiLSTMs perform better on shorter inputs, e.g., ORCAS-NER.

Impact of Gazetteers: Results improve in all cases with external knowledge. While the subtagger and the binary concatenation baselines yield gains compared to the no gazetteer baselines, our CGR-based approach outperforms all of them in all NER tests. This indicates the high effectiveness of our CGR. For LOWNER, using CGR+MoE, MD performance improves by 2.4%, while NER increases 4.7% over the no gazetteer BERT baseline. Low-context data, MSQ-NER and ORCAS-NER, have much lower baseline performance, and benefit greatly from external knowledge. The best MSQ-NER NER model improves 36% over the no gazetteer BiLSTM baseline, while ORCAS-NER increases by 49%. This clearly demonstrates the impact of gazetteer integration.

Effect of Integration Method: CGR outperforms baselines in all NER experiments, showing the effectiveness of a span-aware, contextual representation that is jointly trained with the word-level model. The MoE integration is superior to concatenation in all cases. This is more salient in low context settings, demonstrating that the MoE model can rely on the CGR feature when the syntactic context (CWR) is not discriminative. In some cases baselines actually degrade performance as the model can not effectively balance the experts.

Effect of Two-stage Training: We observe that two-stage training is crucial for BERT, including concatenation models and MoE models, but not for the BiLSTM model. This confirms our hypothesis that the CGR cannot be jointly trained with a large pretrained model. Freezing BERT and then jointly fine-tuning them provides great improvements.

Results on Benchmarks: We applied GEMNET, i.e., BERT using CGR+MoE with two stage training, to the standard benchmarks. We experiment in an uncased setting, and and compare with the reported uncased SOTA (Mayhew et al., 2019). The SOTA uses BERT-CRF, which are the same as our baseline architecture. For comparison, we also reproduce the BERT baseline using our implementation. Results are shown in Table 6. Our models achieve SOTA results in all uncased settings, demonstrating generalization across domains; we improve by 3.9% on WNUT17.

5.3 Per-Class Performance & Error Analysis

We also look at performance across different entity classes to understand the source of our improvements. Table 7 shows relative gains per class, comparing the no gazetteer baseline performance against the best model. Detailed precision/recall values are in Appendix E (Table 16).

The smallest gains are on PER and LOC types, and the largest gains are on products and creative works (CW). This agrees with our hypothesis that these complex entities are the hardest to recognize.

Comparing datasets, increases are much larger on MSQ-NER and ORCAS-NER, confirming the challenges of short low-context inputs, and our models effectiveness in such cases.

	Word Encoder	Gazetter Model	2-stage	LOWNER		MSQ-NER		ORCAS-NER	
				MD	NER	MD	NER	MD	NER
Baseline	BiLSTM	No gazetteer	No	86.5	81.7	62.9	51.4	38.3	27.3
		Subtagger (Liu et al., 2019)	No	**91.0**	86.1	71.1	62.7	56.7	43.8
		Binary Concatenation	No	90.6	87.3	55.7	51.1	41.2	33.3
Ours	BiLSTM	CGR + Concatenation	No	90.8	89.1	84.8	83.2	75.5	73.6
		CGR + Concatenation	Yes	90.7	88.9	85.6	84.1	76.6	74.9
		GEMNET (CGR + MoE)	No	90.6	89.0	**88.7**	**87.3**	**78.1**	**76.3**
		GEMNET (CGR + MoE)	Yes	90.9	**89.3**	86.7	85.6	76.4	75.0
Baseline	BERT	No gazetteer	No	90.5	87.0	65.4	57.3	50.0	37.2
		Subtagger (Liu et al., 2019)	No	90.2	86.3	60.8	53.7	44.8	32.5
		Binary Concatenation	No	87.7	84.2	70.7	60.2	48.7	38.8
Ours	BERT	CGR + Concatenation	No	90.8	87.4	59.8	52.7	46.3	35.7
		CGR + Concatenation	Yes	92.9	91.4	78.4	76.1	63.7	59.0
		GEMNET (CGR + MoE)	No	90.1	86.6	62.7	56.1	47.9	37.2
		GEMNET (CGR + MoE)	Yes	**92.9**	**91.7**	**83.2**	**81.9**	**72.2**	**70.2**

Table 5: MD and NER results (F1 score) on all test sets for models trained on LOWNER.

Method	CoNLL03	WNUT17	OntoNotes
Uncased SOTA	91.0	46.1	88.1
BERT Baseline	89.6	46.9	86.9
GEMNET (BERT)	91.3	50.2	88.0

Table 6: Uncased NER Results (F1 score) on CoNLL03, WNUT17 and OntoNotes v5.0.

Class	LOWNER	MSQ-NER	ORCAS-NER
PER	+1.9	+21.8	+40.1
LOC	+2.2	+37.5	+46.5
GRP	+8.5	+57.3	+57.2
CORP	+12.7	+57.7	+56.5
CW	+10.2	+58.8	+61.4
PROD	+10.7	+64.2	+62.0

Table 7: Relative gains over no gazetteer baseline for each entity class (F1 score) for each dataset.

We also conduct a qualitative error analysis to identify instances where the best non-gazetteer baseline fails, but our model provides correct output. Some examples are shown in Table 8. The baseline often lacks knowledge about complex and long-tail entities, either missing them (#1,6,8 show full or partial MD failure) or misclassifying them (#3-5 show NER errors). Another common trend we observe is baselines incorrectly predicting nested entities within complex entities (#2,10).

5.4 Effect of Gazetteer Coverage

We consider the impact of gazetteer coverage[4] on performance. We hypothesize that training coverage impacts how much the model learns to rely on the gazetteer. To verify this we examine two scenarios: (1) the gazetteer coverage for train and test match (i.e., both high or low); and (2) there is a coverage gap between train and test, e.g., train coverage is 90% but is 25% for test, or vice versa.

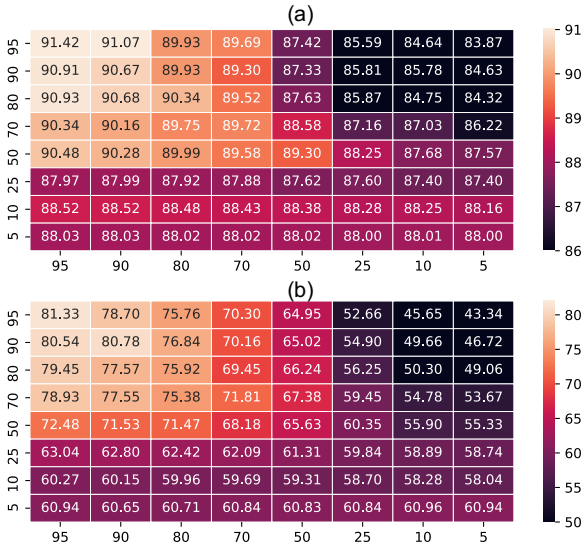

Figure 2: Coverage analysis. X-axis is the testing coverage and Y-axis is the training coverage. (a) shows results for the LOWNER test set, and (b) shows the results for MSQ-NER.

Model and Data: For each train/test set we create gazetteers that have $p\%$ coverage of the set's gold entities, with $p \in \{5, 10, 20, 30, 50, 75, 90, 95\}$. This is achieved by randomly dropping entities. We then train models using each p and evaluate on test sets, using all values of p. This experiment is done using LOWNER and MSQ-NER.

[4]The proportion of entities that are present in the gazetteer

Dataset	Gold Sentence	Entities by Baseline	Entities by Best Model
LOWNER	Example 1: he worked for |linear technology$\|_{CORP}$ and |analog devices$\|_{CORP}$	–	|linear technology$\|_{CORP}$ |analog devices$\|_{CORP}$
	Example 2: his signature piece was |orange blossom special$\|_{CW}$	|orange blossom$\|_{PROD}$	|orange blossom special$\|_{CW}$
	Example 3: it is the last of the |heinlein juveniles$\|_{CW}$	|heinlein juveniles$\|_{GRP}$	|heinlein juveniles$\|_{CW}$
MSQ-NER	Example 4: what is the zip code for |basarbovo$\|_{LOC}$	|basarbovo$\|_{PER}$	|basarbovo$\|_{LOC}$
	Example 5: who is the director of |el reino$\|_{CW}$	|el reino$\|_{GRP}$	|el reino$\|_{CW}$
	Example 6: when was the |nokia 2.2$\|_{PROD}$ invented	|nokia$\|_{CORP}$	|nokia 2.2$\|_{PROD}$
ORCAS-NER	Example 7: |bee-line$\|_{CORP}$ revenue	|bee-line revenue$\|_{CW}$	|bee-line$\|_{CORP}$
	Example 8: |lexus rc 350$\|_{PROD}$ height	|lexus rc$\|_{PROD}$	|lexus rc 350$\|_{PROD}$
	Example 9: how old is |ingross$\|_{PER}$	|ingross$\|_{LOC}$	|ingross$\|_{PER}$
	Example 10: cast of |dr. devil and mr. hare$\|_{CW}$	|dr. devil$\|_{PER}$, |mr. hare$\|_{PER}$	|dr. devil and mr. hare$\|_{CW}$

Table 8: Error analysis examples where baselines fail, but our models provide the correct recognition.

Results: Results are plotted as heatmaps in Figure 2. Best results occur with high train and test coverage, while the worst results fall under high training coverage but low test coverage. When train coverage is low, test coverage has no impact as the model presumably ignores the gazetteer input. Across test coverage values, best results are generally around the diagonal, i.e., matching training coverage. These patterns are identical across datasets, indicating that a train/test coverage gap should be avoided. In practice, if test set coverage cannot be measured, or high coverage is not guaranteed, then using lower training coverage (e.g., 50%) prevents performance degradation in very low test coverage cases.

We also note that the gap between the best and worst result for LOWNER is not huge, showing the impact of sentence context. This gap is much larger for ORCAS-NER, where the model cannot rely on the context. Finally, we note that an alternative dynamic dropout method[5] achieved similar results.

5.5 Performance in a Low-Resource Setting

We also consider the impact of a low-resource setting (limited annotations) on performance, hypothesizing that gazetteers are more helpful in such settings. To verify this, we create random subsets of 5/10/20% of the training data and compare the NER performance of a baseline (BERT-base) vs our best model(BERT+CGR+MoE+2stage) when trained on this data. Results are shown in Table 9.

The results show that gazetteers are always more effective than baseline in low-resource scenarios. Specifically, they improve much faster with less data, achieving close to maximum performance with only 20% of the data.

[5]Gazetteer matches are randomly dropped during training (i.e., random entity dropout).

	LOWNER		MSQ-NER		ORCAS-NER	
Size	Baseline	Ours	Baseline	Ours	Baseline	Ours
5%	74.2	81.2	52.2	60.4	33.1	44.2
10%	78.1	86.7	55.3	74.4	33.7	49.1
20%	81.2	88.7	55.5	81.3	33.9	69.7
100%	87.0	91.7	57.3	81.9	37.2	70.2

Table 9: NER results on the full test set (F1) for comparing a baseline model (BERT, No gazetteer) and GEMNET (BERT + CGR + MoE + 2stage) in low-resource settings using small subsets of the training data.

6 Conclusion

We focused on integrating gazetteers into NER models. We proposed GEMNET, a flexible architecture that includes a Contextual Gazetteer Representation encoder, combined with a novel Mixture-of-Expert gating network to conditionally utilize this information alongside any word-level model. GEMNET supports external gazetteers, allowing the model's knowledge to be updated without retraining.

We also developed new datasets to represent the current challenges in NER. Experimental results demonstrated that our method can alleviate the feature weight under-training issue, achieving significant improvements on our data and a standard benchmark, WNUT17. The datasets we released can serve as benchmarks for evaluating the entity knowledge possessed by models in future work.

Future work involves investigating integration with different model architectures, partial gazetteer matching, and additional entity features.

Acknowledgements

We would like to extend our gratitude to Eugene Agichtein, Alexandre Salle, and Besnik Fetahu for their valuable inputs and discussion during this project. We also thank the anonymous reviewers for their constructive remarks.

References

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153.

Estephe Arnaud, Arnaud Dapogny, and Kevin Bailly. 2020. Tree-gated deep mixture-of-experts for pose-robust face alignment. *IEEE Trans. Biom. Behav. Identity Sci.*, 2(2):122–132.

Sandeep Ashwini and Jinho D. Choi. 2014. Targetable named entity recognition in social media. *CoRR*, abs/1408.0782.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Oliver Bender, Franz Josef Och, and Hermann Ney. 2003. Maximum entropy models for named entity recognition. In *CoNLL*, pages 148–151. ACL.

Gabriel Bernier-Colborne and Philippe Langlais. 2020. Hardeval: Focusing on challenging tokens to assess robustness of NER. In *LREC*, pages 1704–1711. European Language Resources Association.

David Carmel, Ming-Wei Chang, Evgeniy Gabrilovich, Bo-June Paul Hsu, and Kuansan Wang. 2014. Erd'14: entity recognition and disambiguation challenge. In *SIGIR*, page 1292. ACM.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 18 million clicked query-document pairs for analyzing search. *arXiv preprint arXiv:2006.05324*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*. Association for Computational Linguistics.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *SIGIR*, pages 267–274. ACM.

Jiang Guo, Darsh J. Shah, and Regina Barzilay. 2018. Multi-source domain adaptation with mixture of experts. In *EMNLP*, pages 4694–4703. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Abhinav Jain, Vishwanath P. Singh, and Shakti P. Rath. 2019. A multi-accent acoustic model using mixture of experts for speech recognition. In *INTERSPEECH*, pages 779–783. ISCA.

Pratik Jayarao, Chirag Jain, and Aman Srivastava. 2018. Exploring the importance of context and embeddings in neural NER models for task-oriented dialogue systems. *CoRR*, abs/1812.02370.

Daniel Khashabi, Mark Sammons, Ben Zhou, Tom Redman, Christos Christodoulopoulos, Vivek Srikumar, Nicholas Rizzolo, Lev-Arie Ratinov, Guanheng Luo, Quang Do, Chen-Tse Tsai, Subhro Roy, Stephen Mayhew, Zhili Feng, John Wieting, Xiaodong Yu, Yangqiu Song, Shashank Gupta, Shyam Upadhyay, Naveen Arivazhagan, Qiang Ning, Shaoshi Ling, and Dan Roth. 2018. Cogcompnlp: Your swiss army knife for NLP. In *LREC*. European Language Resources Association (ELRA).

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *ACL (1)*, pages 5301–5307. Association for Computational Linguistics.

Zihan Liu, Genta Indra Winata, and Pascale Fung. 2020. Zero-resource cross-domain named entity recognition. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 1–6, Online. Association for Computational Linguistics.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL (1)*. The Association for Computer Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Shervin Malmasi and Mark Dras. 2015. Location mention detection in tweets and microblogs. In *Conference of the Pacific Association for Computational Linguistics*, pages 123–134. Springer.

Stephen Mayhew, Tatiana Tsygankova, and Dan Roth. 2019. ner and pos when nothing is capitalized. In *EMNLP/IJCNLP (1)*, pages 6255–6260. Association for Computational Linguistics.

Svetlana Pavlitskaya, Christian Hubschneider, Michael Weber, Ruby Moritz, Fabian Hüger, Peter Schlicht, and J. Marius Zöllner. 2020. Using mixture of expert models to gain insights into semantic segmentation. In *CVPR Workshops*, pages 1399–1406. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543. ACL.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237. Association for Computational Linguistics.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54.

Shruti Rijhwani, Shuyan Zhou, Graham Neubig, and Jaime G. Carbonell. 2020. Soft gazetteers for low-resource named entity recognition. In *ACL*, pages 8118–8123. Association for Computational Linguistics.

Chan Hee Song, Dawn Lawrie, Tim Finin, and James Mayfield. 2020. Improving neural named entity recognition with gazetteers. *arXiv preprint arXiv:2003.03072*.

Tongshuang Wu, Kanit Wongsuphasawat, Donghao Ren, Kayur Patel, and Chris DuBois. 2020. Tempura: Query analysis with structural templates. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. Luke: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454.

Eun-Suk Yang, Young-Bum Kim, Ruhi Sarikaya, and Yu-Seop Kim. 2016. Drop-out conditional random fields for twitter with huge mined gazetteer. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 282–288.

Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. 2020. Recommendation for new users and new items via randomized training and mixture-of-experts transformation. In *SIGIR*, pages 1121–1130. ACM.

Appendix

A Dataset Details

LOWNER: This dataset is based on Wikipedia, and uses the links as span annotations.

The complete English Wikipedia dump from July 2020 was downloaded. We extracted the articles, which were then parsed to remove markup and extract sentences with their interlinks (links to other articles). This resulted in the extraction of approx. 180 million sentences. We then mapped the interlinks in each sentence to the Wikidata KB then resolved them to our NER taxonomy (in same manner as Appendix B).

Next, we filtered sentences using two strategies. Taking advantage of Wikipedia's well-formed text, we applied a Regex-based NER method to identify sentences containing named entities that were not linked, and removed them. This removes long and high-context sentences that contain references to many entities. Additionally we also removed any sentence where the links could not be resolved to Wikidata entities. This process discards over 90% of the sentences, resulting in approx. 14 million candidate sentences.

This process is very effective at yielding short, low-context sentences. Example sentences are shown in Table 10. The sentences contain some context, but they are much shorter than the average Wikipedia sentence, and usually only contain a single entity, making them more aligned with the challenges listed in Table 1.

We randomly sampled 1.4 million sentences, where entities were tagged using the taxonomy described in Section 3. This forms the complete LOWNER dataset. We created the training, development, and test sets by having the training and dev sets match the CoNLL03 data in size.[6] The remaining items were used to form a very large test set that contains millions of entities not present in the training set.

MSQ-NER: This dataset aims to reflect NER in the QA domain, and is based on the MS-MARCO QnA dataset (v2.1) (Bajaj et al., 2016) which contains over a million questions.

We first templatize the questions by applying an existing NER system (e.g., spaCy) to identify entities in the questions. We then use our gazetteer to map the entities to their NER types to create slotted templates, e.g., "`when did [[iphone]] come out`" becomes "`when did <PROD> come out`". The templates are then aggregated by frequency. This process results in $3,445$ unique question templates.

While the NER system cannot correctly identify many entities, the most frequent templates are reliable. Examples are listed in Table 11.

Finally, we generate MSQ-NER by slotting the templates that have a frequency of $>= 5$ with random entities from the Wikipedia KB with the same class. Each template is slotted with the same number of times it appeared in MS-MARCO in order to maintain the same relative distribution as the original data. This results in $17,868$ questions, e.g., "`when did [[xbox 360]] come out`", which we use as a test set.

ORCAS-NER: To represent the search query domain, we utilize 10 million Bing user queries from the ORCAS dataset (Craswell et al., 2020) and apply the same templatization procedure described above for MSQ-NER. This yields search templates e.g., "`<PROD> price`" and "`<CORP> phone number`", which are used to create annotated queries, e.g., "`[[airpods pro]] reviews`". This process creates $97,324$ unique query templates. We slot these templates according to their frequency, yielding a final dataset of $471,746$ queries. This is our largest, and most challenging, test set. Examples of our templates are listed in Table 12.

B Gazetteer Details and Statistics

We parsed a Wikidata dump from July 2020 and mapped entities to our NER taxonomy (§3). This was done by traversing Wikidata's class and instance relations, and mapping them to our NER classes, e.g., Wikidata's `human` class maps to `PER` in our taxonomy, `song` to `CW`, and so on.

We extracted 1.67 million entities that were mapped to our classes. The distribution of these entities is shown in Table 13.

C Training Details & Hyperparameters

The hyperparameters searching range and the optimal ones we use in Section 5.2, including the results on our created datasets in Table 5 and benchmark results in Table 6, are shown in Table 14 (BiLSTM model) and Table 15 (BERT model). The parameter tuning is performed on the development sets of the respective datasets.

[6] The split ratio between train and dev is about $4:1$.

The design is considered a forerunner to the modern [[food processor]].
The regional capital is [[Oranjestad, Sint Eustatius]].
The most frequently claimed loss was an [[iPad]].
A [[Macintosh]] version was released in 1994.
An [[HP TouchPad]] was prominently displayed in an episode of the sixth season.
The incumbent island governor is [[Jonathan G. A. Johnson]].
A revised edition of the book was released in 2017 as an [[Amazon Kindle]] book.

Table 10: Sample sentences from LOWNER. Gold entities are in brackets.

average retail price of <PROD>
where was <CW> filmed
how many miles from <LOC> to <LOC>
how many kids does <PER> have
when did <GRP> start
when will <CORP> report earnings

Table 11: Sample questions from MSQ-NER. Slots are in angle brackets.

<CW> imdb
best hotels <LOC>
<PER> parents
<PROD> price
<GRP> website
<CORP> customer service

Table 12: Sample search queries from ORCAS-NER. Slots are in angle brackets.

During training we set the gradient norm to be 5.0 to ensure smooth training. We also apply early stopping, halting the training process when we cannot improve performance on the development set during the last 15 epochs.

D CGR Embedding Visualization

As mentioned in §4, given a sentence, we use a gazetteer matcher to extract its representation $\mathbf{M}$. $\mathbf{M}$ is passed to the CGR encoder (i.e., the green paprt in Figure 1) to generate the gazetteer representation, i.e., $\mathbf{h}_{CGR}$. We give all the sentences in MSQ-NER as inputs to the CGR encoder and obtain the averaged embedding vectors of all the gazetteer tags, e.g., B-PER and B-CW. To visualize these gazetteer tags, we apply t-SNE (Maaten and Hinton, 2008) and generate their 2D visualization shown in Figure 3. It is clear that the tags in the same type, e.g., B-PROD and I-PROD, are close to each other. This indicates that our CGR encode can identify the semantic meaning of these tags and provide effective gazetteer representation.

Entity Type (Tag)	#Entries	Examples
PERSON (PER)	799,072	Frank Gray, Steven Jobs
LOCATION (LOC)	430,630	Seattle, Beijing
CORPORATION (CORP)	48,446	Amazon, Sony
GROUPS (GRP)	106,940	Uni. of Cambridge
PRODUCT (PROD)	31,139	TV, Smartphone
CREATIVE-WORK (CW)	256,912	La La Land

Table 13: The distribution of gazetteer entries mapped to each NER class.

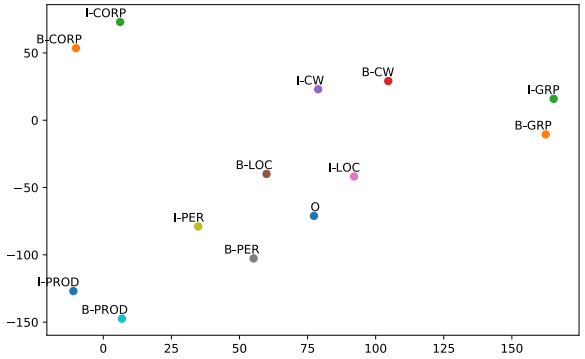

Figure 3: 2D visualization of the CGR (gazetteer tag) representations.

E Additional Results

Some additional detailed results are included in this section.

Table 16 shows detailed precision, recall and F1 scores for each entity class, comparing the no gazetteer baseline model and the best model for each dataset. We note that the worst performance is on products and creative works, as we hypothesized since the entities are much more linguistically complex. These classes achieve the largest increases with our models, which demonstrates that our methods successfully make up the models' weakness in the complex entities challenge.

Parameter	Search Range	LOWNER optimal
BiLSTM input word dimension	[50,200]	50
BiLSTM input charCNN dimension	-	16
BiLSTM input #filters	-	128
BiLSTM input filter size	-	3
BiLSTM hidden	[100, 512]	256
Feedforward #layers	[1,3]	1
Feedforward dimensions	[200,800]	512
Feedforward activation	{linear,tanh,relu}	linear
Matcher dimension	[20,100]	100
Matcher biLSTM hidden	[50, 512]	384
Matcher biLSTM dropout	[0,0.5]	0.1
LearningRate	$[1e-4, 1e-3]$	$1e-3$
Optimizer	{Adam,WAdam}	Adam
Epochs	-	50
Batch size	[16, 32]	32

Table 14: Optimal hyperparameters for BiLSTM-MoE models

Parameter	Search Range	LOWNER optimal	WNUT17 optimal
Feedforward #layers	[1,3]	1	1
Feedforward dimensions	-	768	768
Feedforward activation	{linear,tanh,relu}	linear	linear
Matcher dimension	[20,100]	50	50
Matcher biLSTM hidden	-	384	384
Matcher biLSTM dropout	[0,0.5]	0.1	0.1
1-Stage lr	$[1e-5, 1e-3]$	$3e-4$	$1.5e-3$
1-Stage Optimizer	{Adam,BERT_Adam}	Adam	BERT_Adam
2-Stage lr	$[1e-5, 1e-4]$	$3e-5$	$3e-5$
2-Stage Optimizer	-	BERT_Adam	BERT_Adam
Epochs	-	50	50
Batch size	[16, 32]	25	32

Table 15: Optimal hyperparameters for BERT-MoE models

Dataset	Model	PER			GRP			LOC			CORP			CW			PROD		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
LOWNER	Baseline	93.2	96.3	94.8	80.3	83.1	81.6	91.4	90.4	90.9	82.0	67.7	74.2	74.1	70.9	72.5	54.0	49.7	51.8
	Ours	96.8	96.6	96.7	91.3	88.9	90.1	94.0	92.2	93.1	85.9	87.9	86.9	85.3	80.3	82.7	65.0	60.3	62.5
MSQ-NER	Baseline	68.1	85.0	75.6	28.7	23.2	25.7	61.0	39.6	48.0	50.0	18.3	26.8	23.6	15.1	18.4	46.2	8.9	15.0
	Ours	97.6	97.3	97.4	82.8	83.2	83.0	92.5	79.5	85.5	85.5	83.6	84.5	87.0	69.3	77.2	92.1	69.4	79.2
ORCAS-NER	Baseline	33.9	64.4	44.4	20.9	13.6	16.5	39.5	23.2	29.2	21.3	12.2	15.5	13.9	10.9	12.2	54.0	8.3	14.4
	Ours	78.2	91.9	84.5	69.7	78.2	73.7	81.6	70.6	75.7	66.7	78.2	72.0	72.8	74.3	73.6	91.3	65.7	76.4

Table 16: Per-class performance across entity types. We show the optimal model (Ours) for each dataset, which is BERT+MoE+Two-stage for LOWNER, and BiLSTM+MoE for MSQ-NER and ORCAS-NER. The baselines are BERT, BiLSTM and BiLSTM without gazetteer, respectively.