How Does Beam Search improve Span-Level Confidence Estimation in Generative Sequence Labeling?

Kazuma Hashimoto **Iftekhar Naim** **Karthik Raman**
Google Research, Mountain View
{kazumah, inaim, karthikraman}@google.com

Abstract

Sequence labeling is a core task in text understanding for IE/IR systems. Text generation models have increasingly become the go-to solution for such tasks (e.g., entity extraction and dialog slot filling). While most research has focused on the labeling accuracy, a key aspect – of vital practical importance – has slipped through the cracks: understanding model confidence. More specifically, we lack a principled understanding of how to reliably gauge the confidence of a model in its predictions for each labeled span. This paper aims to provide some empirical insights on estimating model confidence for generative sequence labeling. Most notably, we find that simply using the decoder's output probabilities **is not** the best in realizing well-calibrated confidence estimates. As verified over six public datasets of different tasks, we show that our proposed approach – which leverages statistics from top-k predictions by a beam search – significantly reduces calibration errors of the predictions of a generative sequence labeling model.

1 Introduction

Sequence labeling (e.g., entity extraction) is a fundamental task in building IE/IR systems, such as Web search (Bergsma and Wang, 2007; Fetahu et al., 2021; Guo et al., 2009), QA (Li et al., 2019; Longpre et al., 2021), and goal-oriented dialog (Xie et al., 2022). Prediction confidence is a critical factor for the applications; it is useful to estimate prediction confidence *for each labeled span* in an input text. Beyond direct application (e.g., knowledge distillation (Hinton et al., 2015)), it is crucial to how downstream systems consume the model's output. For example, a high precision system (say a query parser) may choose to only act on high-confidence spans, while falling back or asking for clarifications for low-confidence ones (Xie et al., 2022). Having a well-calibrated model output score – that correlates well with the correctness of the predictions – is important for practical adoption.

There have been increasing attempts to apply text generation models to many NLP tasks (Google, 2023; OpenAI, 2023), since the emergence of pre-trained text generation models like GPT (Radford et al., 2018) and T5 (Raffel et al., 2020). Recent work (Athiwaratkun et al., 2020; FitzGerald, 2020; Raman et al., 2022; Liu et al., 2022) has shown the advantages of the generative approaches for sequence labeling. Given the importance of the span-level confidence estimation and the strength of the generative approaches,

> how do we estimate the confidence of the structured predictions in the text generation?

There has been little work on understanding this.

The natural way to estimate a generative model's confidence for each labeled span is via its corresponding token-level posterior probabilities (Oneata et al., 2021). However, as shown empirically, this approach is not the best; the posterior probabilities arise solely from the top prediction candidate, which may not capture the underlying uncertainty of the complete decoder distribution.

To overcome this limitation, we propose three methods to take full advantage of top-k statistics given by the beam search; **AggSpan** aggregates partial span-level probabilities, **AggSeq** aggregates whole sequence-level probabilities, and **AdaAggSeq** is an adaptive variant of AggSeq, conditioning on complexity of each input. Our experiments, comparing the different confidence estimation methods across six diverse datasets and tasks, show that leveraging the beam-search statistics leads to improving model calibration. Our contributions are summarized as follows:

- we propose methods for span-level confidence estimation in generative sequence labeling,

Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024), pages 62–69

- our extensive experiments show the effectiveness of using the beam-search statistics, and
- we show the robustness of the AdaAggSeq method with a larger beam size.

2 Generative Sequence Labeling

2.1 Task Description

Regardless of what approaches we use, a sequence labeling task T can be formulated as follows:

$$y = f_T(x), \tag{1}$$

where f_T is a task-specific function that takes a text (of n words) $x = [x_1, x_2, \ldots, x_n]$ as an input, and then returns a sequence of m labeled spans $y = [y_1, y_2, \ldots, y_m]$. We assume that the spans are not nested and not overlapped. Such a span y_i is a pair of a contiguous word sequence (or a phrase) s_i and its label ℓ_i: $y_i = (s_i, \ell_i)$.

Here is an example:

x: [FIFA, World, Cup, 2022, in, Qatar],

y: [(FIFA, `ASSOCIATION`), (World Cup, `EVENT`), (2022, `YEAR`), (in, `O`), (Qatar, `COUNTRY`)],

where `ASSOCIATION`, `EVENT`, `YEAR`, and `COUNTRY` are task-specific labels, and `O` is a generic "*outside*" label that is not any of the task-specific labels.

2.2 Prediction by Text Generation

Generative sequence labeling (Vinyals et al., 2015; FitzGerald, 2020) tackles the task by using a conditional text generation model:

$$y = \arg\max_{y'} p_\theta(y'|x), \tag{2}$$

where θ is a set of the model parameters. The most common approach to the model training is teacher forcing (Williams and Zipser, 1989) with human-labeled data, and Equation (2) is approximated by using a beam search (Sutskever et al., 2014).

In the example in Section 2.1, x is represented with a list of words and y with a list of position-sensitive phrase-label pairs, but we can use arbitrary text formats as discussed in Raman et al. (2022). That is, it does not matter which formats we use, as long as we can interpret the outputs.

2.3 Span-level Confidence Estimation

The model's predictions are not always correct, and it is practically useful to inspect the model's prediction confidence (Guo et al., 2017). Specifically, this paper focuses on a span-level confidence score:

$$c_\theta(y_i) \in [0.0, 1.0]. \tag{3}$$

We can use classifier's output (Desai and Durrett, 2020; Hendrycks et al., 2020) with encoder-based token-level classification models (Devlin et al., 2018), but it is less trivial in our case. Malinin and Gales (2021) have studied token-level and sequence-level uncertainty estimation in sequence generation tasks; in contrast, we tackle the confidence estimation *for each labeled span* consisting of a phrase-label pair and its position.

A straightforward approach is to use the conditional probability as follows:

$$c_\theta(y_i) = p_\theta(y_i|x, y_1, \ldots, y_{i-1}), \tag{4}$$

which we call "**span probability**."

Assuming that y_i consists of a sequence of L (subword) tokens $[t_i^1, t_i^2, \ldots t_i^L]$, Equation (4) is computed as follows:

$$\prod_{j=1}^{L} p_\theta(t_i^j|x, y_1, \ldots, y_{i-1}, t_i^1, \ldots, t_i^{j-1}). \tag{5}$$

Previous work investigated various methods to aggregate partial confidence scores (e.g., token-level scores in ASR systems (Oneata et al., 2021) and pixel-level scores in image segmentation (Mehrtash et al., 2020)), but we have observed that Equation (5) robustly works as a solid baseline.

3 Beam Search-based Estimation

Equation (4) only uses the probability values regarding the top-1 candidate by the beam search. Therefore, it is not taken into account what labeled spans are likely predicted in other sequence-level outputs in the generative labeling process.

We study two confidence estimation methods that reflect statistical information given by the beam search, inspired by the effectiveness of using the beam search on sequence-level knowledge distillation (Kim and Rush, 2016; Wang et al., 2020).

3.1 Aggregated Span Probability

We consider incorporating broader contexts to estimate plausibility of generating y_i given x:

$$c_\theta(y_i) = p_\theta(y_i|x) = \sum_z p_\theta(y_i|x, z)p_\theta(z|x), \tag{6}$$

where z is a generated context before predicting y_i, and $z = [y_1, \ldots, y_{i-1}]$ is such an example.

Equation (6) is not tractable, and we compute its estimation by the beam search:

$$c_\theta(y_i) = \frac{\sum_{z_\mathcal{B}} p_\theta(y_i|x, z_\mathcal{B}) p_\theta(z_\mathcal{B}|x)}{\sum_{z_\mathcal{B}} p_\theta(z_\mathcal{B}|x)}, \tag{7}$$

where $z_\mathcal{B}$ is a unique context that exists in top-k candidates generated by the beam search. Note that, if there is only one unique context in the k candidates, Equation (7) is reduced to Equation (4). We call the method "**aggregated span probability**."

3.2 Aggregated Sequence Probability

Next, we consider using whole sequence-level information to define $c_\theta(y_i)$, which is a missing ingredient in Equation (7). More specifically, we aggregate the sequence-level probabilities such that the sequences contain y_i:

$$c_\theta(y_i) = \sum_{\hat{y}} p_\theta(\hat{y}|x), \tag{8}$$

where $\hat{y}$ is a complete output sequence generated by the model, containing y_i.

We use the beam search for its estimation:

$$c_\theta(y_i) = \frac{\sum_{\hat{y}_\mathcal{B}} p_\theta(\hat{y}_\mathcal{B}|x)}{\sum_{j=1}^{k} p_\theta(y^{(j)}|x)}, \tag{9}$$

where $\hat{y}_\mathcal{B}$ is a $\hat{y}$ that is in the top-k candidates, and $y^{(j)}$ is the j-th best candidate. Intuitively, Equation (9) counts how frequently y_i appears in the top-k candidates, by weighting the counts with the sequence-level probabilities. We call the method "**aggregated sequence probability**." Note that this method is useful only with $k > 1$, because $k = 1$ always results in $c_\theta(y_i) = 1.0$.

Adaptive strategy The larger value of k we use, the more output variations this method takes into account, which is expected to be reasonable when the output space is complex. In contrast, it makes less sense to use a large value of k for an output with only a few non-O spans. To alleviate the potential issue, we propose an adaptive alternative by replacing the constant k in Equation (9) with an adaptive value $k' \in [2, k]$. We measure the complexity of the output space by counting the number of non-O spans in the top-1 candidate, and set

$$k' = \max(2, \min(a + b, k)), \tag{10}$$

where a is the counted number and b is a hyper-parameter. We call the method "**adaptive aggregated sequence probability**."

	Train	Validation	Test	Non-O spans
ATIS	4,478	500	893	3.4
SNIPS	13,084	700	700	2.6
mTOP	15,667	2,235	4,386	1.7
MIT-R	6,845	789	1,516	2.0
NER	14,987	3,466	3,684	1.8
CHUNK	8,936	1,844	2,012	12.0

Table 1: Statistics of the six datasets.

3.3 Estimation of AggSpan and AggSeq

In Sections 3.1 and 3.2, we used the beam search to obtain estimation of Equations (6) and (8), respectively. We explain the estimation process.

Aggregated span probability: We consider the effects of the beam size k. In particular, with $k \gg 1$, Equation (7) is expressed as follows:

$$\begin{aligned} &\frac{\sum_{z_\mathcal{B}} p_\theta(y_i|x, z_\mathcal{B}) p_\theta(z_\mathcal{B}|x)}{\sum_{z_\mathcal{B}} p_\theta(z_\mathcal{B}|x)} \\ \approx &\frac{\sum_{z} p_\theta(y_i|x, z) p_\theta(z|x)}{\sum_{z} p_\theta(z|x)} \\ = &\sum_{z} p_\theta(y_i|x, z) p_\theta(z|x), \end{aligned} \tag{11}$$

because of $\sum_z p_\theta(z|x) = 1$, resulting in Equation (6).

Aggregated sequence probability: Similarly, we can express Equation (9) as follows:

$$\begin{aligned} \frac{\sum_{\hat{y}_\mathcal{B}} p(\hat{y}_\mathcal{B}|x)}{\sum_{j=1}^{k} p(y^{(j)}|x)} &\approx \frac{\sum_{\hat{y}} p_\theta(\hat{y}|x)}{\sum_{y'} p_\theta(y'|x)} \\ &= \sum_{\hat{y}} p_\theta(\hat{y}|x), \end{aligned} \tag{12}$$

because of $\sum_{y'} p_\theta(y'|x) = 1$, resulting in Equation (8).

4 Reliability of Confidence Estimation

We expect that, the higher a confidence score is, the more accurate the prediction will be, and vice versa. To evaluate how reliable the confidence scores are, we adapt a widely-used metric, Expected Calibration Error (ECE) (Guo et al., 2017; Mehrtash et al., 2020; Desai and Durrett, 2020).

For each evaluation example x in a dataset, we have a prediction y and its corresponding ground-truth annotation y^*. A predicted span in y is treated as correct if it agrees with y^*; more concretely, y^* needs to contain a span whose position, phrase, and label are exactly the same as those of the predicted

	ATIS (F1: 0.942 ± 0.003)		SNIPS (F1: 0.930 ± 0.014)		mTOP (F1: 0.906 ± 0.006)	
	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}
Span	0.014 ± 0.000	0.036 ± 0.001	**0.018** ± 0.003	0.039 ± 0.006	0.026 ± 0.001	0.062 ± 0.002
AggSpan	0.014 ± 0.000	0.036 ± 0.001	**0.018** ± 0.003	**0.038** ± 0.006	**0.025** ± 0.000	0.060 ± 0.002
AggSeq	**0.011** ± 0.003	**0.020** ± 0.007	0.023 ± 0.004	0.039 ± 0.007	**0.025** ± 0.004	**0.041** ± 0.009
	MIT-R (F1: 0.802 ± 0.010)		NER (F1: 0.890 ± 0.010)		CHUNK (F1: 0.960 ± 0.004)	
	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}
Span	0.046 ± 0.003	0.119 ± 0.009	0.011 ± 0.001	0.075 ± 0.004	0.023 ± 0.001	0.026 ± 0.001
AggSpan	0.045 ± 0.003	0.118 ± 0.009	0.010 ± 0.001	0.074 ± 0.004	0.022 ± 0.001	0.026 ± 0.001
AggSeq	**0.011** ± 0.002	**0.023** ± 0.006	**0.007** ± 0.003	**0.030** ± 0.008	**0.021** ± 0.001	**0.020** ± 0.001

Table 2: ECE scores ($k = 5$) on the ATIS, SNIPS, mTOP, MIT-R, NER, and CHUNK test sets. The lower a score is, the better it is. The value range of the metrics is in $[0.0, 1.0]$. For reference, F1 scores are also shown.

span. We collect all the predicted spans from all the evaluation examples, resulting in a set of N predicted spans in total.

We then assign a group index m ($1 \leq m \leq M$) for each predicted span whose confidence score falls into the m-th confidence bin $(\frac{m-1}{M}, \frac{m}{M}]$. An ECE metric is defined as follows:

$$\text{ECE} = \frac{1}{N} \sum_{m=1}^{M} N_m |\text{ACC}_m - \text{MC}_m|, \quad (13)$$

where N_m is the number of spans in the m-th group, and ACC_m and MC_m are the accuracy and mean confidence of the group, respectively. We then use the following two ECE metrics:
- $\textbf{ECE}_{\text{ALL}}$ evaluates all the predicted spans,
- $\textbf{ECE}_{\text{NO}}$ evaluates only non-O spans.

5 Experiments

We conduct experiments to empirically compare the three methods: **Span** (Equation (4)), **AggSpan** (Equation (7)), and **AggSeq** (Equation (9)), by setting $k = 5$ for the beam search, and $M = 10$ for the reliability estimation. We then evaluate the adaptive AggSeq (**AdaAggSeq**) with $k = 10$.

5.1 Datasets, Text Format, and Model

To perform the evaluation on diverse datasets and tasks with a strong model, we strictly follow experimental settings in a previous study (Hashimoto and Raman, 2022). The following datasets are used: **ATIS** (Price, 1990), **SNIPS** (Coucke et al., 2018), **mTOP** (Li et al., 2021), **MIT-R**,[1] **NER** (Tjong Kim Sang and De Meulder, 2003), and **CHUNK** (Tjong Kim Sang and Buchholz, 2000).

- **ATIS**: slot-filling in travel assistance,
- **SNIPS**: slot-filling in virtual assistance,
- **mTOP**: semantic parsing in voice assistance,
- **MIT-R**: semantic parsing in dining assistance,
- **NER**: CoNLL 2003 named entity recognition,
- **CHUNK**: CoNLL 2000 syntactic chunking.

Table 1 shows the number of sentence-level examples, and the average number of (per sentence) annotated non-O spans in the validation sets.

We use the "sentinel+tag (SI)" format proposed in Raman et al. (2022), to represent the input and output texts in our experiments. This format is known to be effective in avoiding hallucinations in the text generation. To run our experiments, we use the pre-trained mT5 "base" (Xue et al., 2021) in the T5X code base (Roberts et al., 2022). Details of the fine-tuning process are described in Appendix.

5.2 Results and Discussions

We run all the experiments five times, and the average scores are reported along with the standard deviation values.

Effects of the beam search Table 2 shows the comparison between the three methods. "Span" already has a good calibration ability as expected, thanks to the use of the large pre-trained model (Desai and Durrett, 2020). We then see that either "AggSpan" or "AggSeq" is consistently better than "Span," which shows the effectiveness of using the beam search statistics.

Case Study We have inspected the results for more intuitive interpretation. One observation is that "AggSeq" tends to better reflect the model's uncertainty when predictions contradict with their ground-truth annotations. Table 3 shows such an example from the MIT-R validation set, where estimated confidence scores are shown for each

[1]https://groups.csail.mit.edu/sls/downloads/.

Input	do you have listings of diners in the area
Gold	(do, `O`), (you, `O`), (have, `O`), (listings, `O`), (of, `O`), (diners, `Cuisine`), (in, `O`), (the, `O`), (area, `Location`)
Top-5	1: (do, `O`), (you, `O`), (have, `O`), (listings, `O`), (of, `O`), (diners, `Cuisine`), (in the area, `Location`)
	2: (do, `O`), (you, `O`), (have, `O`), (listings, `O`), (of, `O`), (diners, `Cuisine`), (in, `O`), (the, `O`), (area, `Location`)
	3: (do, `O`), (you, `O`), (have, `O`), (listings, `O`), (of, `O`), (diners, `Cuisine`), (in, `Location`), (the, `O`), (area, `Location`)
	4: (do, `O`), (you, `O`), (have, `O`), (listings, `O`), (of, `O`), (diners, `O`), (in the area, `Location`)
	5: (do, `O`), (you, `O`), (have, `O`), (listings, `O`), (of, `O`), (diners, `Cuisine`), (in, `O`), (the, `O`), (area, `O`)
Span	(do, `O`)$_{\mathbf{0.99}}$, (you, `O`)$_{\mathbf{0.99}}$, (have, `O`)$_{\mathbf{0.99}}$, (listings, `O`)$_{\mathbf{0.99}}$, (of, `O`)$_{\mathbf{0.99}}$, (diners, `Cuisine`)$_{\mathbf{0.99}}$, (in the area, `Location`)$_{\mathbf{0.87}}$
AggSpan	(do, `O`)$_{\mathbf{0.99}}$, (you, `O`)$_{\mathbf{0.99}}$, (have, `O`)$_{\mathbf{0.99}}$, (listings, `O`)$_{\mathbf{0.99}}$, (of, `O`)$_{\mathbf{0.99}}$, (diners, `Cuisine`)$_{\mathbf{0.98}}$, (in the area, `Location`)$_{\mathbf{0.86}}$
AggSeq	(do, `O`)$_{\mathbf{1.0}}$, (you, `O`)$_{\mathbf{1.0}}$, (have, `O`)$_{\mathbf{1.0}}$, (listings, `O`)$_{\mathbf{1.0}}$, (of, `O`)$_{\mathbf{1.0}}$, (diners, `Cuisine`)$_{\mathbf{0.93}}$, (in the area, `Location`)$_{\mathbf{0.63}}$

Table 3: Confidence estimation for an ambiguous span. Erroneous spans are shown with underlines.

	ATIS		SNIPS		mTOP	
	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}
AggSeq	0.021 ± 0.003	0.035 ± 0.009	0.036 ± 0.007	0.059 ± 0.011	0.044 ± 0.005	0.068 ± 0.012
AdaAggSeq	$\mathbf{0.009} \pm 0.002$	$\mathbf{0.016} \pm 0.007$	$\mathbf{0.016} \pm 0.003$	$\mathbf{0.028} \pm 0.005$	$\mathbf{0.013} \pm 0.003$	$\mathbf{0.022} \pm 0.005$
	MIT-R		NER		CHUNK	
	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}	ECE_{ALL}	ECE_{NO}
AggSeq	0.020 ± 0.002	0.028 ± 0.012	0.015 ± 0.003	0.042 ± 0.011	0.023 ± 0.001	0.023 ± 0.001
AdaAggSeq	$\mathbf{0.010} \pm 0.002$	$\mathbf{0.020} \pm 0.003$	$\mathbf{0.005} \pm 0.001$	$\mathbf{0.026} \pm 0.003$	$\mathbf{0.022} \pm 0.001$	$\mathbf{0.022} \pm 0.001$

Table 4: Evaluation of "AggSeq" and "AdaAggSeq" ($k = 10$) with $b = 3$ for MIT-R and $b = 1$ for the rests.

of the estimation methods. We can see that "AggSeq" assigns the lowest confidence score to the `Location` label, because the "(in the area, `Location`)" span appears only in the first and fourth candidates out of the top-5 candidates.

Effects of the beam size k Next, we investigate the effects of the beam size k when using "AggSeq." Table 4 shows the results with $k = 10$, and we can see that $k = 10$ performs worse than $k = 5$ by comparing the scores with those in Table 2. Only the CHUNK results are comparable; this is presumably because the output space of the CHUNK task is considered to be the most complex as evidenced in Table 1.

As expected, "AdaAggSeq" helps resolve the issue discussed in Section 3.2, and the improved scores are even better than those of "AggSeq" with $k = 5$ (except for CHUNK). The use of k' in Equation (10) makes "AggSeq" more robust, and future work is to investigate how to better measure the complexity of each example.

Which one should we use? One natural question is which method we want to use in practice; we recommend "AdaAggSeq" based on our empirical results, if one can use a validation set to determine the value of b. Otherwise, "AggSeq" with $k = 5$ is a good choice because it robustly works across the six different datasets. However, the beam search introduces non-negligible computational costs when performing inference on billions of inputs. We can use "Span" in such a case.

Applicability to blackbox models Recently, not all the large pre-trained models are published; GPT-4 (OpenAI, 2023) and PaLM 2 (Google, 2023) are such examples. In case the token-level probabilities are not visible but the whole sequence-level probabilities are available, (Ada)AggSeq has advantages of being used along with the blackbox models.

6 Conclusion

We have investigated effective ways of estimating span-level confidence in generative sequence labeling, and shown that the top-k statistics help improve reliability of the estimation. We believe that our work provides a basis for future work like learning to improve the confidence reliability and using the confidence scores in real applications.

Acknowledgments

We thank Krishna Srinivasan for his providing useful comments to improve the drat. We also appreciate fruitful discussions with Leonid Teverovsky and Kiran Yalasangi, about potential use cases of the confidence estimation methods.

Limitations

Choice of pre-trained models We used a multilingual variant (Xue et al., 2021) of T5 (Raffel et al., 2020) to test the span-level confidence estimation methods, motivated by strong empirical results in previous work (Raman et al., 2022; Liu et al., 2022). However, all the equations in this paper are based on the very basic idea in Equation (2), and it is not specific to the T5 model architecture. For example, the conditional text generation can be implemented with decoder-only models like GPT (Radford et al., 2018). We can use different types of pre-trained text generation models (BART (Lewis et al., 2020), GPT, T5, etc.).

Choice of input/output text formats We used a particular input/output text format among a variety of possible formats investigated in previous work (FitzGerald, 2020; Raman et al., 2022), to minimize concern about hallucinations in the text generation process. However, all the equations in this paper are not specific to any of the existing text formats. We can thus adapt the estimation methods to other text formats, as long as we can interpret the outputs as in Section 2.1.

Application to more complex tasks We targeted sequence labeling tasks where labeled spans are not nested as mentioned in Section 2.1; in other words, there are no overlaps between the labeled spans. An interesting extension of our work is to adapt the confidence estimation methods to more complex tasks like those in Barnes et al. (2022) and Liu et al. (2022).

Access to prediction probability In the end of Section 5.2, we discussed the applicability of our methods to the blackbox models. We expect that probability-like scores are available with the predictions, but it would be possible that some APIs only provide predictions without such scores. Therefore, it is another important line of work to consider reliability of the predictions in such restricted scenarios.

References

Ben Athiwaratkun, Cicero Nogueira dos Santos, Jason Krone, and Bing Xiang. 2020. Augmented natural language for generative sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 375–385, Online. Association for Computational Linguistics.

Jeremy Barnes, Laura Ana Maria Oberländer, Enrica Troiano, Andrey Kutuzov, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, and Erik Velldal. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Shane Bergsma and Qin Iris Wang. 2007. Learning noun phrase query segmentation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 819–826, Prague, Czech Republic. Association for Computational Linguistics.

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*.

Alexandre de Brébisson and Pascal Vincent. 2016. The Z-loss: a shift and scale invariant classification loss belonging to the Spherical Family. *arXiv preprint arXiv:1604.08859*.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 295–302, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer enhanced named entity recognition for code-mixed web queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1677–1681, New York, NY, USA. Association for Computing Machinery.

Jack FitzGerald. 2020. STIL - Simultaneous Slot Filling, Translation, Intent Classification, and Language Identification: Initial Results using mBART on MultiATIS++. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 576–581.

Google. 2023. PaLM 2 Technical Report.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 267–274, New York, NY, USA. Association for Computing Machinery.

Kazuma Hashimoto and Karthik Raman. 2022. GROOT: Corrective Reward Optimization for Generative Sequential Labeling. *arXiv preprint arXiv:2209.14694*.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. Mtop: A comprehensive multilingual task-oriented semantic parsing benchmark.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics.

Tianyu Liu, Yuchen Jiang, Nicholas Monath, Ryan Cotterell, and Mrinmaya Sachan. 2022. Autoregressive Structured Prediction with Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 993–1005.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7052–7063, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Andrey Malinin and Mark Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations*.

Alireza Mehrtash, William Wells, Clare Tempany, Purang Abolmaesumi, and Tina Kapur. 2020. Confidence calibration and predictive uncertainty estimation for deep medical image segmentation. *IEEE Transactions on Medical Imaging*, PP:1–1.

Dan Oneata, Alexandru Caranica, Adriana Stan, and Horia Cucu. 2021. An evaluation of word-level confidence estimation for end-to-end automatic speech recognition. pages 258–265.

OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*.

Patti Price. 1990. Evaluation of spoken language systems: The atis domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Karthik Raman, Iftekhar Naim, Jiecao Chen, Kazuma Hashimoto, Kiran Yalasangi, and Krishna Srinivasan. 2022. Transforming Sequence Tagging Into A Seq2Seq Task. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11856–11874.

Adam Roberts, Hyung Won Chung, Anselm Levskaya, Gaurav Mishra, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, Curtis Hawthorne, Aitor Lewkowycz, Alex Salcianu, Marc van Zee, Jacob Austin, Sebastian Goodman, Livio Baldini Soares, Haitang Hu, Sasha Tsvyashchenko, Aakanksha Chowdhery, Jasmijn Bastings, Jannis Bulian, Xavier Garcia, Jianmo Ni, Andrew Chen, Kathleen Kenealy, Jonathan H. Clark, Stephan Lee, Dan Garrette, James Lee-Thorp, Colin Raffel, Noam Shazeer, Marvin Ritter, Maarten Bosma, Alexandre Passos, Jeremy Maitin-Shepard, Noah Fiedel, Mark Omernick, Brennan Saeta, Ryan Sepassi, Alexander Spiridonov, Joshua Newlan, and Andrea Gesmundo. 2022. Scaling Up Models and Data with `t5x` and `seqio`. *arXiv preprint arXiv:2203.17189*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive Learning Rates with Sublinear Memory Cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 Shared Task Chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

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.

Oriol Vinyals, Ł ukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a Foreign Language. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Fei Huang, and Kewei Tu. 2020. Structure-level knowledge distillation for multilingual sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3317–3330, Online. Association for Computational Linguistics.

Ronald J. Williams and David Zipser. 1989. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2):270–280.

Tian Xie, Xinyi Yang, Angela S. Lin, Feihong Wu, Kazuma Hashimoto, Jin Qu, Young Mo Kang, Wenpeng Yin, Huan Wang, Semih Yavuz, Gang Wu, Michael Jones, Richard Socher, Yingbo Zhou, Wenhao Liu, and Caiming Xiong. 2022. Converse: A tree-based modular task-oriented dialogue system.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498.

Appendix

We fine-tune the pre-trained model for each dataset separately, based on the negative log-likelihood loss (Hashimoto and Raman, 2022). We use the Adafactor optimizer (Shazeer and Stern, 2018), along with Z-loss regularization (de Brébisson and Vincent, 2016), where a constant learning rate of 0.001 is used. The training is run for upto 2500 steps (evaluating checkpoints after every 100 steps). We select the best checkpoint per the F1 score on the validation set of each dataset. The T5X code base is publicly available.[2]

[2] `https://github.com/google-research/t5x`.