Predicting Entity Salience in Extremely Short Documents

Benjamin L. Bullough, Harrison Lundberg, Chen Hu, Weihang Xiao
Amazon / USA
{bullough, lundbh, chenwho, weihanx}@amazon.com

Abstract

A frequent challenge in applications that use entities extracted from text documents is selecting the most salient entities when only a small number can be used by the application (e.g., displayed to a user). Solving this challenge is particularly difficult in the setting of extremely short documents, such as the response from a digital assistant, where traditional signals of salience such as position and frequency are less likely to be useful. In this paper, we propose a lightweight and data-efficient approach for entity salience detection on short text documents. Our experiments show that our approach achieves competitive performance with respect to complex state-of-the-art models, such as GPT-4, at a significant advantage in latency and cost. In limited data settings, we show that a semi-supervised fine-tuning process can improve performance further. Furthermore, we introduce a novel human-labeled dataset for evaluating entity salience on short question-answer pair documents.

1 Introduction

Entity salience (ES) is a natural language understanding task concerned with determining which entities mentioned in a passage of text are most salient to the passage. Salience refers to the centrality of an entity to the content of a text rather than the intrinsic importance of the entity beyond the text or its relevance to the perspective of a particular reader (Gamon et al., 2013). If entities are to be automatically extracted from the text, entity recognition and linking is performed before applying an entity salience model. The role of the ES model is to score the entities so they can be filtered (or ranked) by downstream applications.

In the example in Figure 1, there are three entities extracted from the question-answer (Q/A) pair: Popsicle, Frank Epperson and San Francisco. The entity salience task is to classify the entities in the Q/A pair as salient or non-salient. The ground truth labels for this example indicate that Popsicle and Frank Epperson are salient while San Francisco is non-salient. Note that the extraction and linking of the entities is done by a separate entity recognition and linking model and is not considered part of the entity salience task.

Question: Who invented the Popsicle?
Answer: The Popsicle was invented by Frank Epperson, an 11-year-old from San Francisco.
Entities:

- **Name**: Popsicle, **Salience**: True
- **Name**: Frank Epperson, **Salience**: True
- **Name**: San Francisco, **Salience**: False

Figure 1: Entity Salience Task Example

Compared to long studied NLP tasks such as named entity recognition and linking (Sevgili et al., 2020), ES has received less attention in the literature. Even less studied is the problem of determining salience in the context of very short documents. However, very short documents have become an increasingly important type of data in many online applications such as social media posts, customer reviews and question answering systems, and determining entity salience plays a crucial role in many applications where focusing on a subset of entities from a document is required.

In extremely short documents, many of the signals that are useful features for determining salience in longer documents, such as position, frequency and co-occurrence patterns are likely to be absent or attenuated (Sharma and Li, 2019). In this setting, a semantic understanding of the document and the entities is more critical. Large language models and their ability to represent text through dense embeddings are a natural fit in this situation.

In this paper, we propose to model the salience of an entity as the similarity of its embedding to the

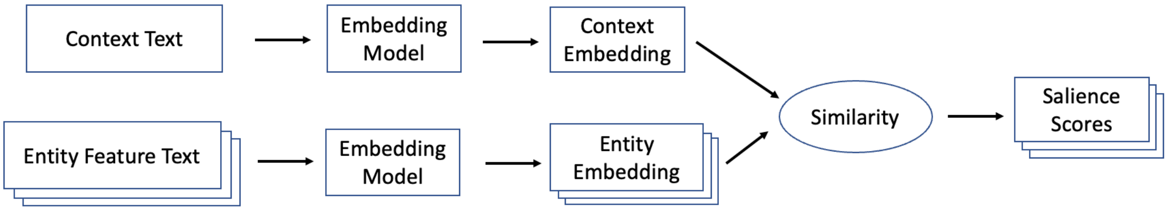


Figure 2: **Entity salience model with biencoder embedding model**

embedding of the overall text passage, where each embedding is generated by a common sentence encoder. Specifically, this paper makes the following contributions:

1. We propose an approach for ES prediction suitable for extremely short documents using a sentence encoder applied to both the document and text representations of each entity. This approach can leverage powerful, pre-trained language models to generate the embeddings and requires no labeled training data. Despite being lightweight, data-efficient and low-latency, it achieves competitive performance with respect to more heavyweight models such as GPT-4.
2. We describe how pre-trained sentence encoders can be further improved in limited data settings by fine-tuning on unlabeled in-domain data using a two step semi-supervised training approach where a cross-encoder teacher model is bootstrapped from pseudo-labels derived from the pre-trained sentence encoder.
3. We create a novel human-labeled dataset for ES evaluation on extremely short documents, which to the best of our knowledge is the first labeled entity salience dataset that focuses on short documents. To facilitate further research, we open source our dataset[1].

2 Entity Salience Model

2.1 Biencoder model

We model the salience of an entity extracted from a passage as the similarity (i.e. cosine similarity) of the entity embedding to the context passage embedding. Given a text embedding function f_{emb} and a similarity function f_{sim}, we calculate the salience score $s_{salience}$ of an entity represented by entity feature text x_{ent} with respect to a context passage $x_{context}$ as,

$$s_{salience} = f_{sim}(f_{emb}(x_{ent}), f_{emb}(x_{contex})) \quad (1)$$

To get a binary salience classification $c_{salient}$, we apply a simple threshold t to the salience score, which can be tuned to control the trade-off between type I and II errors.

$$c_{salient} = s_{salience} > t \quad (2)$$

We calculate the passage and entity embeddings using a sentence embedding Transformer network (Vaswani et al., 2017; Reimers and Gurevych, 2019). One of the advantages of such a model is the flexibility in choosing the text that will represent the entity and be the input to the sentence embedding model (i.e., the entity feature text). A minimal approach would be to use only the entity name or mention text, but other sources of information include the entity description from a knowledge base or the entity type labels from a named entity recognition (NER) system. Text from various sources can be concatenated to form the entity feature text. Figure 2 illustrates the proposed modeling approach, which is similar to other works that have used used bi-encoder models for text classification and scoring tasks such as Schopf et al. (2022); Gao et al. (2021); Reimers and Gurevych (2019).

2.2 Semi-supervised fine-tuning with cross-encoder teacher

While pre-trained sentence encoder models can be used with the above model of entity salience, performance can be improved by adapting the encoder to the ES task and the target domain. If labeled examples are available, they can be used directly for supervised fine-tuning, however in practice there are often few labeled examples readily available and avoiding expensive labeling efforts (in terms

[1] https://github.com/amazon-science/entity-salience-short-documents

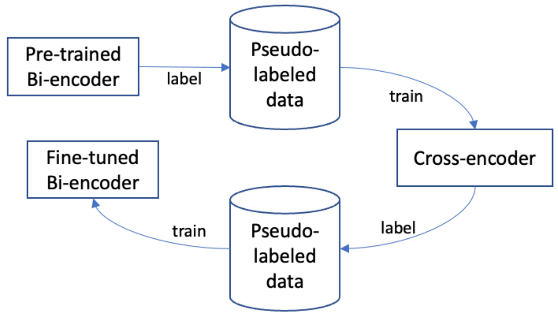


Figure 3: **Semi-supervised fine-tuning training flow**

of time and money) is desirable. To this end, we describe a semi-supervised approach to fine-tuning that does not require any labeled training examples (and only a few hundred labeled evaluation examples) but instead bootstraps a cross-encoder teacher model using the initial pre-trained model to label training examples. This teacher model is then used to fine-tune the final biencoder model. Our approach is inspired by the iterative process of bootstrapping models described in Liu et al. (2021) and the data augmentation strategies described in Thakur et al. (2021).

As illustrated in Figure 3, the first step is to initialize the biencoder ES model with a pre-trained sentence embedding model such as SBERT (Reimers and Gurevych, 2019) or GTE (Li et al., 2023). This initial model is used to generate entity salience scores for each context-entity pair in the dataset.

Next, we initialize a cross-encoder model (Reimers and Gurevych, 2019) with a pre-trained language model such as RoBERTa (Liu et al., 2019), and fine-tune it with Binary Cross Entropy (BCE) loss using the scores generated in the previous step [2]. We use the fine-tuned model to relabel the training dataset.

Finally, we fine-tune a pre-trained biencoder model using our training dataset with labels generated by the cross-encoder[3]. We train using Multiple Negatives Ranking Loss (MNRL) with presumed positive examples (Henderson et al., 2017). MNRL creates in-batch negatives by re-pairing the entities with context from other pairs. Previous work has shown that MNRL is superior to cross-entropy loss for training sentence embedding models (Reimers, 2022). To select the positive examples, we set a threshold on the scores from the cross-encoder and tune this threshold as a hyper-parameter.

3 WikiQA-Salience dataset

Existing ES datasets (Gamon et al., 2013; Dunietz and Gillick, 2014; Wu et al., 2020) focus on longer documents such as web pages and news articles (with many hundreds of words) rather than the short texts that are the focus of this work. In this section, we describe the creation of a dataset, WikiQA-Salience, for evaluating entity salience on extremely short question-answer pair passages.

We leveraged the WikiQA dataset as a starting point to create a new ES dataset from publicly available data. The WikiQA corpus is an answer sentence selection (AS2) dataset where the questions are derived from query logs of the Bing search engine, and the answer candidates are extracted from Wikipedia (Yang et al., 2015). The examples are Q/A pairs in natural language with full-sentence (non-factoid) answers, which resemble the type of responses provided by conversational assistants.

We first selected the Q/A pairs in the corpus where the answer is labeled as correctly answering the question (positive pairs). Then we applied the ReFinED named entity resolution model (Ayoola et al., 2022) to the combined question-answer text to extract named entities[4].

We augment each entity with the name, description and aliases of the entity from WikiData. Since WikiData descriptions are typically extremely brief, we further augment the entities in the dataset with more detailed information from Wikipedia pages (wherever these are available) including the Wikipedia summary (i.e., the first section of the page) and the first 100 noun-phrases from the arti-

[2] We experimented with both binary cross entropy (BCE) loss and mean squared error (MSE) loss and found that BCE worked better for training the cross-encoder. This aligns with the observations of Liu et al. (2021), which concluded that BCE is a "temperature-sharpened version of MSE, which is more tolerant towards numerical discrepancies"

[3] See Appendix A.1 for an explanation of our choice to target a biencoder architecture for deployment.

[4] The extraction of entities is performed as a distinct preprocessing step before the entity salience model by a state-of-the-art entity extraction model (ReFinED), which was developed independently (prior to our work). Since our work focuses only on identifying the degree of salience of entities, an entity being tagged does not imply salience. The usage scenarios we envision for our entity salience model include automated extraction of entities. We consider this aspect of the data generation process to add a degree of realism to the evaluation data. The manual salience annotation step performed after the automated entity extraction provided an opportunity for human review of the entities, and erroneously extracted entities would be expected to be tagged as low-salience by annotators.

cle extracted with the spaCy NLP library.

Ground truth labels were generated by crowd workers on the Amazon Mechanical Turk platform who rated the relevance of each entity to the Q/A pair it was extracted from on a three level scale (“High”, “Moderate”, “Low”). Five independent passes of annotation were performed for each entity. The finished dataset consists of 687 annotated Q/A pairs with the linked entity data from ReFinED, entity details from WikiData, and the (5-pass) crowd worker salience ratings. To aggregate the multiple annotator passes, we take the median rating (after mapping to numeric values), which unlike majority vote considers the inherent ordering of the labels. The 687 Q/A pairs contain 2113 entities (unique at the Q/A pair level), and the mean length of the question-answer text is just 190.6 characters and 32.9 words. The distribution of the ES labels is significantly skewed towards salient entities with 1089 rated “High”, 535 rated “Moderate” and 489 rated “Low”. For the purpose of using the labels in a binary classification task, in this work we map High and Moderate ratings to "salient" and Low to "non-salient". The inter-rater agreement of the binary labels measured by the Fleiss’ kappa score is 0.230, which indicates "fair agreement" between the annotators (Hartling, 2012). Additional details on the construction of the dataset can be found in Appendix A.6.

4 Experiments

In the first set of experiments, we evaluate four pre-trained models in combination with several sets of entity text features. The four pre-trained models are composed of two model sizes selected from two families of sentence embedding models: all-MiniLM-v2 (SBERT) (Reimers and Gurevych, 2019; Wang et al., 2020) and GTE (Li et al., 2023). Table 1 lists the pre-trained sentence embedding models used in this paper along with the number of parameters, word embedding dimension and maximum sequence length for each model. We access the base models from the HuggingFace Model Hub with the SentenceTransformers library and evaluate performance on the WikiQA-Salience dataset described above.

We treat the concatenated question and answer text for entries in the dataset as the context for predicting salience. For entity text features, we consider the entity name (`name`), Wikidata entity description (`desc`), the first section of the entity Wikipedia page (`fs`) and the first 100 noun-phrases from Wikipedia (`np`), as well as combinations of these. Table 2 gives the details for each set of entity text features used.

We measure performance using macro averaged F1 score (macro-F1) where the operating point of each model has been tuned to maximize the macro-F1 score. Our reason for using macro averaging in this case, rather than the more conventional binary F1 score, is that the classes are highly imbalanced and we consider classification accuracy on the negative (“non-salient”) class as important as classification on the positive class (“salient”). Macro-F1 balances these considerations and makes the metric invariant to the assignment of labels to the classes.

In the second set of experiments, we look at the performance benefits from fine-tuning the model. As a source of unlabeled training data, we use 23,843 Q/A pairs published on the Alexa Answer crowd sourcing platform (AlexaAnswers, 2023). We also use 500 manually labeled examples from Alexa Answers as our validation dataset for selecting the best model checkpoint during fine-tuning. We evaluate these models using the same test set and metrics as in the first set of experiments. Additional details about the model training process are included in Appendix A.4.

As a baseline, we compare our method to the entity salience prediction methods used in Wu et al. (2020), which were adapted from Dunietz and Gillick (2014). We focus on the position and frequency oriented features that previous works found to be useful but adapt them to the context of extremely short documents. We use the character position of the start of the first mention (`first-mention-position`) as a feature instead of the sentence of the first mention. We also use the rank order of the first mention (`first-mention-order`) and the number of times that the entity is explicitly mentioned (`mention-count`). We use a random forest classification model which we train and evaluate using three-fold cross-validation on the WikiQA-Salience dataset.

To provide additional benchmarks for assessing model performance, we also evaluate the following:

- `predict-positive`: Always predict majority class label (i.e. “salient”). This uninformed classifier provides a lower bound for performance.
- `human-annotator`: Response of a single

model name	model parameters	word embedding dimension	max seq length
all-MiniLM-L6-v2	22.7 M	384	256
all-MiniLM-L12-v2	33.4 M	384	128
gte-small	33.4 M	384	512
gte-base	109.5 M	768	512

Table 1: **Base biencoder models**

feature set name	feature set description
`desc`	Entity description from Wikidata
`name`	Entity name from Wikidata
`fs`	First section of Wikipedia
`np`	First 100 noun-phrases from Wikipedia
`name-desc`	`name` + `desc`
`name-desc-fs`	`name` + `desc` + `fs`
`name-desc-np`	`name` + `desc` + `np`

Table 2: **Entity text feature sets**

randomly selected human rater (1-pass) evaluated against the ground truth labels based on the consensus of a 5-pass annotation process. This serves as a benchmark for what a model with human level abilities would score on this dataset. Given the variation in human labeling, we do not see 100% agreement in the human ratings. It serves to highlight the difficulty and the inherent subjectivity of the task.

- `gpt-4-zero-shot-name`: Prompted GPT-4 LLM model (OpenAI, 2023) with zero-shot inference (using entity name as feature text). Additional details about this model including the prompt used are provide in Appendix A.5.
- `crossencoder-roberta-*`: Two cross-encoder models based on RoBERTa, with and without the first section of the entity Wikipedia page in the feature text. The former is the teacher model for the fine-tuned biencoders.

5 Results

5.1 Overall performance compared to baselines

Table 3 summarizes the performance of our ES models in the context of the baselines. The pre-trained gte-small sentence transformer model using entity name as the feature text (`gte-small-name`) achieves an macro-F1 score of 70.3%, which far exceeds the 54.7% obtained by the best baseline using position and frequency features. However, `gte-small-name` lags GPT-4 on this task by 1.8% (absolute) and lags human performance by 5.2%. Fine-tuning the model improves macro-F1 by 1.5% while a similar amount of improvement (1.8%) comes from adding the Wikidata entity description to the entity name (still using the pre-trained model). Fine-tuning and adding entity descriptions in combination (`gte-small-finetuned-name-desc`) adds 3.0%, which exceed zero-shot GPT-4[5] by 1.2% and is only 2.3% below human annotator performance. We see that this biencoder model also achieves parity with the cross-encoder teacher model.

model	macro-F1
predict-positive	0.435
first-mention-position (baseline)	0.505
+ first-mention-order	**0.547**
+ mention-count	0.538
gte-small–name	0.703
gte-small-finetuned–name	0.718
gte-small–name-desc	0.721
gte-small-finetuned–name-desc	**0.733**
crossencoder-roberta-base–name-desc	0.716
crossencoder-roberta-base–name-desc-fs	0.725
gpt-4-zero-shot–name	0.721
human-annotator	**0.756**

Table 3: **Comparison to baselines (macro-F1)**

5.2 Pre-trained models

We compare the performance of four pre-trained models (two sizes of two model families) using four different source of entity feature text (individually) and three additive combinations. These experiments show how the different sources of information about entities as well as the size and quality of the text embedding model impact performance on the entity salience task. The results

[5] We find that the latency and cost of using a generative LLM for this tasks is substantially higher than our approach - roughly a two orders of magnitude difference. See Appendix A.2 for a comparison of the latency and cost.

feature-set	desc	name	fs	np	name-desc	name-desc-fs	name-desc-np
base-model							
all-minilm-l6-v2	**0.681**	0.692	0.713	0.713	0.707	0.713	0.716
all-minilm-l12-v2	0.673	0.686	0.711	0.708	0.707	0.708	0.709
gte-small	0.675	**0.703**	0.726	0.721	**0.721**	0.723	**0.725**
gte-base	0.659	0.702	**0.732**	**0.725**	0.717	**0.726**	0.721

Table 4: **Pre-trained models with different feature sets (macro-F1)**

in Table 4 show that using the very concise Wikidata entity descriptions alone perform worse then using the entity name alone while using the first section of the Wikipedia entry performed best. Using a summary of Wikipedia articles based on the first 100 noun-phrases performed about the same as using the first section without any further summarization. The combination of entity name and Wikidata description was much better then either alone and is nearly as good as the first Wikipedia section.

We see that the newer GTE family of models consistently outperforms the older SBERT (all-MiniLM) models. However there does not appear to be a consistent improvement from using a larger model size within the same model family.

5.3 Fine-tuned models

feature-set-name	name	name-desc	name-desc-fs
student-model			
all-minilm-l6-v2	0.692	0.707	0.713
all-minilm-l6-v2-ft	0.715	0.728	0.728
all-minilm-l12-v2	0.686	0.707	0.708
all-minilm-l12-v2-ft	0.712	0.721	0.717
gte-small	0.703	0.721	0.723
gte-small-ft	**0.718**	**0.733**	**0.734**
gte-base	0.702	0.717	0.726
gte-base-ft	0.704	0.724	0.724

Table 5: **Pre-trained vs fine-tuned on different feature sets (macro-F1). Fine-tuned models are indicated with a "ft" suffix.**

Next, we measure the impact of fine-tuning on all four pre-trained sentence encoder models with three progressively larger sets of entity features[6]. Table 5 shows the results with fine-tuning relative to the pre-trained versions of each model. We see that fine-tuning improves performance in almost every case. The benefits of fine-tuning appear to be the greatest where the pre-trained model-feature set combinations are weakest. The clear gap between the all-MiniLM and GTE model families is substantially reduced after fine-tuning.

Appendix A.3 describes additional ablation experiments that show the performance of different cross-encoder models and their impact as teacher models for fine-tuning the final biencoder.

6 Discussion

6.1 Diminishing returns from using longer entity descriptions

The results from our experiments show that using more details about entities improves ES predictions, but we observe diminishing marginal returns to increasing the amount of feature text. For example using the gte-small model, adding very brief Wikidata descriptions to the entity name adds 1.8% (absolute) to the macro-F1 score, but adding the full Wikipedia summary only adds another 0.2%. This trend is seen with other models and with the fine-tuned versions as well. This observation is important given the additional compute and latency costs associated with processing longer sequences, and it suggests using information-dense descriptions that maximize the amount of information per token processed.

6.2 Strong performance of pre-trained models

Another finding is the relatively strong performance of using pre-trained sentence transformer models. While we see consistent gains from fine-tuning these models, the gains are relatively modest. This speaks to the strong cross-task and cross-domain generalization of these models and is consistent with the findings of Wang et al. (2021) who found that the best “out-of-the-box” models (which are fully trained with available supervised data including STS and NLI datasets) are hard to beat for most tasks.

6.3 Impact of embedding dimension, max sequence length and model size

It is also interesting that the most consistent pattern in performance of the pre-trained models across different sets of features is that the GTE models outperform the all-MiniLM models while there is relatively little difference within each family of

[6] We use the same cross-encoder teacher model with entity name, Wikidata description and the first section of the entity Wikipedia page as the entity feature text

models. The gte-base model has over three times as many parameters and twice the word embedding dimension as the gte-small (see Table 1) suggesting that model capacity is not a limitation for these models in the ES task.

One significant difference between the model families is that the GTE models have double (or more) the maximum sequence length. While this could certainly be a factor when we use larger amounts of entity feature text, the performance gap is also clearly present when only the entity name is used, which suggest another factor, such as the data and tasks with which they are trained, is responsible for the difference.

7 Deployed Application

Our use case is for a conversational assistant, where entity salience is used to identify the most important entities in the current question-answer interaction. The most salient entity is used for retrieving explorable content to show on the screen accompanying the spoken answer. Our baseline system relied on simple heuristics of entity count and position in the context with respect to a predicted answer span. We selected the `gte-small-finetuned-name` model to deploy due to runtime constraints on the availability of features and tuned the operating point, using a labeled dataset, to primarily improve the recall of salient entities while not harming the precision.

We deployed the new model alongside the baseline system in an A/B test and monitored performance for two weeks. Our key online performance indicators were the percentage of question-answer interactions where we identified at least one salient entity (salient entity coverage) and the click-through-rate of explorable content shown on screen (CTR). Compared to the baseline system, the entity salience model increased salient entity coverage by 25.7% (relative) and the CTR of explorable content increased 2%.

8 Related Work

A few previous studies have looked at the topic of entity salience. Most have focused on longer documents and found that structural features (such as position, use in title, etc.) or statistical features (frequency of occurrence) are the most useful features for their models (Gamon et al., 2013; Dunietz and Gillick, 2014; Wu et al., 2020). Some more recent works have also incorporated word or entity embeddings as features (Ponza et al., 2018; Xiong et al., 2018). Contemporaneous work by Asgarieh et al. (2024) explored entity salience detection in news articles by fine-tuning pre-trained transformer models with classification heads that use contextualized entity embeddings.

A closely related research area to ES is keyword extraction - selecting a small number of words (or phrases) from a document which can concisely describe most important topics in the document. ES can be conceived of as a keyword extraction task where the set of keywords to be considered is limited to the named entities in the document.

While there is a large and diverse body of literature on keyword extraction techniques (Hasan and Ng, 2014), prior methods typically employ different combinations of statistical, graph-based, and embedding-based features (Rose et al., 2010; Campos et al., 2020; Mihalcea and Tarau, 2004; Wan and Xiao, 2008; Bougouin et al., 2013; Wang et al., 2015). Embedding based methods of keyword extraction generally work by comparing the similarly of keyword embedding to a passage embedding. This category of methods is most closely related to the work described in this paper. Notable, examples include EmbedRank (Bennani-Smires et al., 2018) and KeyBERT (Grootendorst, 2023). Also, Sharma and Li (2019) uses an unsupervised embedding based approach to generate (noisily) labeled examples which are used to train a model.

9 Conclusion

In this work, we propose a model for entity salience that works in the context of extremely short documents and introduce a new dataset for evaluating entity salience based on WikiQA. We show that this simple model can perform well in conjunction with pre-trained sentence transformers. We also demonstrate a data efficient approach to fine-tuning the model that achieves performance on-par with the far larger GPT-4 model on the entity salience task, while achieving far lower latency and cost, and is within a few percentage points of human performance on our dataset.

References

AlexaAnswers. 2023. Alexa answers website. `https://alexaanswers.amazon.com/about`.

Eliyar Asgarieh, Kapil Thadani, and Neil O'Hare. 2024. Scalable detection of salient entities in news articles.

Tom Ayoola, Shubhi Tyagi, Joseph Fisher, Christos Christodoulopoulos, and Andrea Pierleoni. 2022. ReFinED: An efficient zero-shot-capable approach to end-to-end entity linking. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*. Association for Computational Linguistics.

Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Adrien Bougouin, Florian Boudin, and B. Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. *International Joint Conference on Natural Language Processing*, page 543–551.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. YAKE! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Jesse Dunietz and Daniel Gillick. 2014. A new entity salience task with millions of training examples. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*. Association for Computational Linguistics.

Michael Gamon, Tae Yano, Xinying Song, Johnson Apacible, and Patrick Pantel. 2013. Identifying salient entities in web pages. In *Proceedings of the 22nd ACM international conference on Conference on information and knowledge management - CIKM '13*. ACM Press.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Maarten Grootendorst. 2023. Maartengr/keybert: v0.8.

Lisa Hartling, editor. 2012. *Validity and inter-rater reliability testing of quality assessment instruments*. Number no. 12-EHC039-EF in AHRQ publication. Agency for Healthcare Research and Quality, Rockville, MD. "March 2012.". - Includes bibliographical references. - Description based on online resource; title from PDF title page (viewed June 6, 2012).

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yunhsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning.

Fangyu Liu, Yunlong Jiao, Jordan Massiah, Emine Yilmaz, and Serhii Havrylov. 2021. Trans-encoder: Unsupervised sentence-pair modelling through self- and mutual-distillations.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2021. Show your work: Scratchpads for intermediate computation with language models.

OpenAI. 2023. Gpt-4 technical report.

Marco Ponza, Paolo Ferragina, and Francesco Piccinno. 2018. Swat: A system for detecting salient wikipedia entities in texts.

Nils Reimers. 2022. Sentencetransformers documentation. `https://github.com/UKPLab/sentence-transformers`. Accessed: 2023-10-31.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents.

Tim Schopf, Daniel Braun, and Florian Matthes. 2022. Evaluating unsupervised text classification: Zero-shot and similarity-based approaches.

Ozge Sevgili, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. 2020. Neural entity linking: A survey of models based on deep learning.

Prafull Sharma and Yingbo Li. 2019. Self-supervised contextual keyword and keyphrase retrieval with self-labelling.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge.

Kexin Wang, Nils Reimers, and Iryna Gurevych. 2021. Tsdae: Using transformer-based sequential denoising auto-encoderfor unsupervised sentence embedding learning. *arXiv preprint arXiv:2104.06979*.

Rui Wang, Wei Liu, and Chris Mcdonald. 2015. Corpus-independent generic keyphrase extraction using word embedding vectors.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models.

Chuan Wu, Evangelos Kanoulas, Maarten de Rijke, and Wei Lu. 2020. Wn-salience: A corpus of news articles with entity salience annotations. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 2095–2102. European Language Resources Association.

Chenyan Xiong, Zhengzhong Liu, Jamie Callan, and Tie-Yan Liu. 2018. Towards better text understanding and retrieval through kernel entity salience modeling.

Yi Yang, Wen tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

A Appendix

A.1 Deployment considerations and selection of biencoder architecture

Biencoders are traditionally considered the most efficient option when comparisons need to be made between a large number of items, as in retrieval tasks. When a smaller number of comparisons are required, as in a re-ranking task, cross-encoders are often used because of their ability to model token level interactions between pairs of items with the self-attention mechanism (Reimers and Gurevych, 2019).

In this paper, we focus on using a biencoder architecture for the entity salience task, despite its similarity to the re-ranking tasks. One practical reason is that using a biencoder allows us to cache or pre-compute entity embeddings, which reduces compute requirements and decreases latency. Additionally, we found that we were actually able to achieve similar performance with a fine-tuned biencoder architecture compared to a cross-encoder teacher model (as noted in the results section). However, the overall model bootstrapping approach and the dataset presented in this paper could easily be adapted to produce a final cross-encoder student model if desired.

A.2 Latency and cost comparison

We show that GPT-4 achieves performance similar to our model with no fine-tuning, thus it could be considered as an alternative to the approach we describe in this paper in some situations. However, the latency of the GPT-4 API (at the present time) is on the order of several seconds for our task, which is too slow for our conversational voice assistant use case.

One of the benefits of the simplicity of our approach is the relatively low latency and cost. While any comparisons are likely to become quickly dated, given the rapid changes in computing infrastructure and LLM technology, here we provide some "back-of-the-envelop" calculations to illustrate the stark difference in latency and cost between our approach and a "state-of-the-art" LLM. (The cost and latency estimates below are current as of October 2024).

To make the comparison as fair as possible, we make some changes to the evaluated GPT-4 setup. We simplify the prompt in Table 9 to remove the explanations and numeric score from the output, which substantially reduces the number of output tokens that contribute the most to latency and cost. We also use the `gpt-3.5-turbo-0125` model from OpenAI as a more competitive option in terms of cost and latency. Over the WikiQA-Salience dataset, we observed a median latency of 1.12 seconds for each example. The median number of input and output tokens were 188 and 49, respectively. Assuming an average cost per million tokens of $0.5 (input) and $1.5 (output), the average cost per entity salience example is $0.0001675.

In contrast, when running our proposed method with the GTE-small model on a AWS EC2 `g4dn.xlarge` GPU host, we observe an average latency of 0.013 seconds per example. Assuming, an hourly on-demand rate of $0.526 for a `g4dn.xlarge` instance, sequential processing of each example, and a constant demand that fully utilizes the host, the average cost per example with our approach is a mere $0.00000189. (In a realistic deployment, actual costs could vary based on opportunities to process multiple requests in parallel as well the need to scale for peak traffic loads.)

While simplified, this analysis shows that the proposed entity salience approach using sentence embeddings is roughly two orders of magnitude lower in latency and two orders of magintude more cost efficient compared to using an LLM such as GPT-3.5.

A.3 Comparing different teacher models

We experiment with using different teacher models with different sets of features. (Table 6 describes the base teacher models used.) In addition to using two cross-encoder models, we experiment with a simplification of the two-step fine-tuning procedure, where we used the initial scores from the all-MiniLM-L6-v2 biencoder to directly in fine-tuning the biencoder, removing the cross-encoder from the process. We also attempt to make the cross-encoder more robust to specific entity featurizations by training with multiple copies of each training example where each uses a different featurization (labled “multiple”).

Table 7 shows the performance on the test dataset of cross-encoders models fine-tuned from RoBERTa-base using different sets of entity text features (at training and inference time). We see that performance improves with more extensive entity descriptions, as was the case with the biencoder models (see section 5.2).

Table 8, shows the performance of student bi-

model name	model type	model parameters	max seq length
RoBERTa-base	cross-encoder	124.6M	512
DistillRoBERTA	cross-encoder	82.1M	512
all-MiniLM-L6-v2	biencoder	22.7M	256

Table 6: **Base teacher models**

model	
crossencoder-roberta-base–name	0.691
crossencoder-roberta-base–name-desc	0.716
crossencoder-roberta-base–name-desc-fs	0.725
crossencoder-roberta-base-multiple–name-desc-fs	0.724

Table 7: **Crossencoder teacher performance (macro-F1)**

encoder models fine-tuned with different teacher models. In all cases the base student model is the gte-small model using entity name and Wikidata descriptions as feature text. We see do not find a consistent improvement in the performance of the student model from using a larger teacher model or one with access to enhanced features.

A.4 Training parameter details

Crossencoder training:

- weight decay: 0.01
- batch size: 16
- epochs: 1
- loss function: Binary Cross Entropy

Biencoder training:

- weight decay: 0.01
- batch size: 16
- epochs: 1
- loss function: Multiple Negatives Ranking Loss

A.5 Prompt for GPT-4 Baseline

The prompt template used with GPT-4 is shown in Table 9. The predictions were generated using the OpenAI ChatCompletion API on July 14, 2023 with the temperature parameter set to zero.

We prompt the model to provide an explanation for the rating of each entity before generating the categorical rating and the numeric score as a form of "scratchpad" (Nye et al., 2021) or "chain-of-thought" (Wei et al., 2022) reasoning. We found that the numeric score (using a threshold for salience of greater than 5) was better for predicting entity salience than the categorical rating and use this in our results.

A.6 Details of WikiQA-Salience dataset construction and labeling

A.6.1 The WikiQA dataset

The WikiQA corpus (Yang et al., 2015) is an answer sentence selection (AS2) dataset where the questions are derived from query logs of the Bing search engine, and the answer candidate are extracted from Wikipedia.

As described in the WikiQA download page, "the WikiQA corpus is a new publicly available set of question and sentence pairs, collected and annotated for research on open-domain question answering. In order to reflect the true information need of general users, we used Bing query logs as the question source. Each question is linked to a Wikipedia page that potentially has the answer. Because the summary section of a Wikipedia page provides the basic and usually most important information about the topic, we used sentences in this section as the candidate answers. With the help of crowdsourcing, we included 3,047 questions and 29,258 sentences in the dataset, where 1,473 sentences were labeled as answer sentences to their corresponding questions." Table 10 shows examples of question-answer pairs from WikiQA.

We assessed that the WikiQA corpus would be a suitable starting point for offline evaluation of ES models (in the context of question/answer pairs from a voice assistant) because it had the following properties:

- The examples are question answer pairs (QA domain)
- The questions are posed in natural language
- The answers are short sentences (with potentially multiple entities) rather than “factoid” answers.

teacher-features teacher-model	name	name-desc	name-desc-fs	multiple
biencoder	0.732	0.734	0.740	0.738
distilroberta-base	0.731	0.740	0.730	0.736
roberta-base	0.731	0.740	0.734	0.734

Table 8: **Comparing biencoder student models fine-tuned with different teacher models (macro-F1)**

- The question are “self-contained” and do not explicitly reference a “context” document.

A.6.2 Subsampling and Data Preparation

We start with the full WikiQA corpus in the WikiQA.tsv file. This file contains 29208 question/answer (QA) pairs with one or more candidate answers for each question. The candidate answers in the corpus are assigned a binary label based on whether the they answer the question. We select for use only the QA pairs with positive labels (1469) since these are most similar to the answers served to Alexa users. We combine the question and answer into a single passage of text by concatenating the question and answer text. We join the question and answer with just two spaces separating them, avoiding the inclusion of additional punctuation which might influence the subsequent entity extraction process.

A.6.3 Extracting linked entities with ReFinED and augmenting extracted entities with WikiData

We next apply the ReFinED entity detection and linking model to each combined text passage to derive the candidate entities that are part of each QA pair. Of the initial 1469 QA pairs, 282 have no named entities mentions and 246 have only one. Since these are not likely to provide useful examples to assessing the capacity of a model to select the most salient entities in a QA pair, we exclude these examples.

We exclude examples with seven or more entity mentions (110 cases) and also exclude cases where there is not more than one unique entity and remove duplicate questions from the dataset, giving preference to the question/answer pairs with the most entities. After filtering based on the number and uniqueness of entities and questions we are left with 696 example QA pairs.

For each entity in the dataset we retrieve the following additional information (when available) from WikiData based on the WikiData entity ID produced by the ReFinED entity linker: entity name, entity description, entity aliases. We used the pywikibot library to assist with this task. This information is stored with each entity in the dataset.

Table 12 shows the statistics for the length of the question-answer text (i.e. the context). Table 13 shows the distribution of the number of entities in the Q/A pairs.

A.6.4 Wikipedia summaries

Since Wikidata descriptions are typically extremely brief, we further augment the entities in the dataset with more detailed information from Wikipedia pages (wherever these are available). We include the Wikipedia page summary from the first section of the page, the first 100 non-phrases from the article and the first 100 key phrases obtained using two different key phrase extraction algorithms.

We use the wikipedia python package to download Wikipedia pages from the internet for each entity. We extract noun phrases using spacy with the `en_core_web_sm` model. For extracting key phrases we also use spacy and rake-nltk.

A.6.5 Annotation with Amazon Mechanical Turk (mTurk)

Given the dataset of question/answer pairs with entities linked, we want a ground truth rating of the salience of each entity that we can use to evaluate the performance of a model on this task. To obtain this final piece of information for our evaluation dataset, we rely on human annotation (i.e. ground truth labeling).

Amazon Mechanical Turk (mTurk) is a cloud-based service which allows “requesters” with “human intelligence tasks” (HITs) to submit tasks to be performed by “workers” who are paid a monetary reward for each task they complete. The requester provide the tasks via a user defined HTML form which includes the instructions, the information about the specific task and the mechanism to collect the data from the worker. The requester also specifies the size of the reward.

To prepare the dataset for annotation, where each entity mention will be labeled independently for its salience (i.e. relevance to the QA pair), we “explode” each row (containing all the entity mentions

```
prompt_template = """
You are an editor for a newspaper who has to identify the most critical pieces of
↪  information when writing the headline for an article.

For this task you are given a question-answer pair as Context and a list of
↪  entities from the text.  Read the Context given in triple backticks and rate
↪  how salient each entity is to the Context.  Before answering provide a short
↪  justification for your answer.

Provide a salience score in the range of 0 to 10 where 0 is least salient and 10 is
↪  the most salient.

Provide a categorical rating from the following options:
    High - The entity is strongly related to the main point of the question-answer
    ↪  pair or is the answer itself.
    Moderate - The entity is related to the question-answer pair but it is not the
    ↪  most important part.
    Low - The entity not related or is only tangentially or superficially related
    ↪  to the question-answer pair.

Countries (especially in reference to nationality) are frequently incidental to the
↪  answer and are most often "Low" salience unless directly related to the
↪  question.

Give your answer as valid JSON in the following format:
[
{{
"entity": <entity_name>,
"explanation": <explanation of the rating>,
"rating": <rating>,
"score": <score>,
}}
]

Context: ```{context_str}```

List of entities: {entity_str}

Answer:
"""
```

Table 9: **GPT-4 prompt template for entity salience task**

for a QA pair) into multiple rows with one row for each entity mention. This yields a dataset with 2573 entity mentions.

From this dataset we selected 5 QA pairs with 19 entity mentions for "gold" annotation by members of the research team. These were selected non-randomly by the investigator with the goal of choosing examples that contained both salient and non-salient examples in each sentence. Seven members of the research team completed the annotation task in which they rated each entity on a three level scale of relevance to the QA pair: Low, Moderate, High. The results of the gold annotation were included in "control" tasks used to evaluate the accuracy of crowd workers. In the four cases where there was substantial variance in label assigned by the annotators, the "gold" answer was given as a set of correct answers (e.g. "Low", "Moderate". Out of the 19 gold entity mentions, 6 were chosen to additionally serve as qualification tasks. These were selected due to the uniformity (i.e. low variance) of the labels given by the gold annotators (to ensure a minimum of ambiguity in assessing potential crowd workers) and to cover both salient and non-salient examples. The gold examples were combined with the other non-gold examples to form the the final set of 2573 ES annotation tasks.

The labeling task was defined in a templated HTML form that displayed the task instructions, the question text, the answer text, the entity mention text, the resolved entity name from WikiData, and the entity description from WikiData. The workers were asked to select a relevance rating for the entity of Low, Moderate or High and optionally to leave a comment about the task. The full text of the annotation instructions is shown in Table 11.

Question	Answer
how big is bmc software in houston, tx	Employing over 6,000, BMC is often credited with pioneering the BSM concept as a way to help better align IT operations with business needs.
how much is jk rowling worth	The 2008 Sunday Times Rich List estimated Rowling's fortune at £560 million ($798 million), ranking her as the twelfth richest woman in the United Kingdom.
how long was frank sinatra famous	Beginning his musical career in the swing era with Harry James and Tommy Dorsey , Sinatra found unprecedented success as a solo artist from the early to mid-1940s after being signed to Columbia Records in 1943.

Table 10: **Example QA pairs from WikiQA**

Before working on the task workers were required to complete the six qualification tasks with an accuracy of at least 5 out of 6 correct. Workers were eligible to qualify if they were from an English speaking country (US, GB, AU, NZ, CA) and had completed at least 100 previous HITs with an approval rate of 95% or higher. Up to 300 workers were allowed to attempt qualification. 21 qualified before the maximum number of qualification attempts was reached. (This took on the order of 15 minutes after publishing the qualification tasks.)

The workers were offered a total compensation of $0.10 per task ($0.06 reward + $0.04 bonus) and expected to take at least 20 seconds for each task. The 2573 HITs were split into four batches of up to 650 tasks. To increase the quality of the final labels used in evaluation, each task was assigned to 5 different workers, resulting in five independent labels for each task, which can be aggregated to get a "consensus" label (as described below). The batches were completed from 15-16 December 2022, with each batch being completed within 1-2 hours of being published. On average, the workers took more than 60 seconds per task.

A.6.6 Post-processing and analysis

After the mTurk annotation jobs were completed, the results of each job were merged into a single dataset for post-processing and analysis. Nine tasks were missing annotation due to task submission errors related to missing entity descriptions. The nine QA pairs with unlabeled entities were removed from the dataset leaving 687 fully annotated QA pairs.

While we include the full set of annotator ratings for each entity, in the final dataset we also include two aggregations of the ratings which make it easier to work with the data. First we convert the ratings levels to numeric values ("low": 0, "moderate": 1 and "high": 2) and normalize them to be in the range [0, 1]. Then we take the median and mean average of the normalized numeric rating. (Note that we do not use "majority vote" because the rating values have an inherent ordering that should be considered. With five pass annotation, the median is a close analog to selection by majority vote, but it handles "ties" between two levels rationally.) Table 14 shows the distribution of entities median salience ratings in the final dataset.

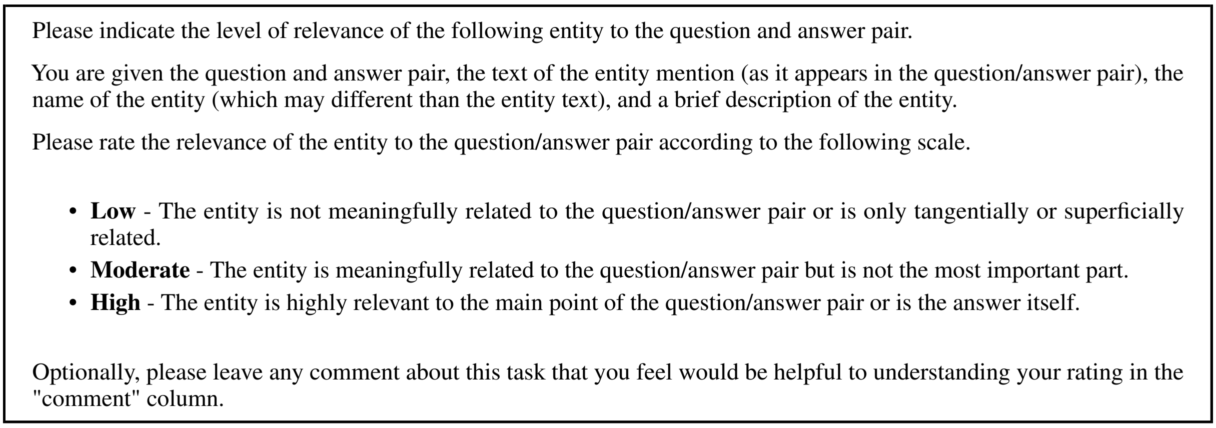
Using the mean rating value for binary classification requires settings a threshold below which the score is considered to indicate a non-salient rating. A reasonable choice for this would be 0.25 (half way between "Low" and "Moderate"). However, a choice of 0.33 results in a minimum number of disagreements with using the median (where 0.0/"Low" maps to "non-salient"). Ultimately, the choice of which aggregation and threshold to use can be made by the user of the dataset and both aggregations and the raw ratings are included.

The inter-rater agreement of the binary labels (derived from median annotator ratings) measured by the Fleiss' kappa score is .230, which indicates "fair agreement" between the annotators (Hartling, 2012). To get an additional indication of inter-rater agreement and establish a goalpost for human-level performance on the ES task, we compare the individual ratings to that of the (median) average rating. Over 500 runs of Monte Carlo simulation, for each entity in the dataset we randomly select a single rating and compare it to the consensus rating. We perform a binary comparison, whether both labels agree that it is relevant or non-relevant, which allows us to calculate accuracy for a single label vs the consensus label. Using this process we measure the human level accuracy as 82.2% (std=0.68).

A.6.7 Finished dataset

The dataset includes the following fields, which can be joined with the original WikiQA data using the QuestionID and SentenceID fields:

- QuestionID: QuestionID from the original WikiQA dataset


Please indicate the level of relevance of the following entity to the question and answer pair.

You are given the question and answer pair, the text of the entity mention (as it appears in the question/answer pair), the name of the entity (which may different than the entity text), and a brief description of the entity.

Please rate the relevance of the entity to the question/answer pair according to the following scale.

- **Low** - The entity is not meaningfully related to the question/answer pair or is only tangentially or superficially related.
- **Moderate** - The entity is meaningfully related to the question/answer pair but is not the most important part.
- **High** - The entity is highly relevant to the main point of the question/answer pair or is the answer itself.

Optionally, please leave any comment about this task that you feel would be helpful to understanding your rating in the "comment" column.

Table 11: **mTurk Annotation Guidelines**

	mean	**std**	**min**	**max**
characters	190.6	68.4	49	589
words	32.9	11.2	8	89

Table 12: **Context size**

number of entities	**count**
2	281
3	188
4	125
5	71
6	22
Total	687

Table 13: **Distribution of number of entities**

- SentenceID: SentenceID from the original WikiQA dataset
- entities: a list of entity objects

Each entity object contains the following fields:

- text: the mention text
- category: the coarse mention type (from ReFinED)
- predicted-entity-types: the predicted entity types
- wikidata-entity-id: the WikiData entity ID
- el-score: the ReFinED entity linking model confidence score
- start-char: the start character of the mention text within the passage
- end-char: the end character of the mention text within the passage
- backend: the name of the entity linking model (i.e. "refined")

median rating	**count**
High	1089
Moderate	535
Low	489
Total	2113

Table 14: **Distribution of ground truth labels**

- wikidata-entity-name: the canonical name of the entity in WikiData
- wikidata-entity-description: a short textual description of the entity from WikiData
- wikidata-entity-aliases: a list of aliases for the entity from WikiData
- gt-rating-mean: the mean normalized numeric rating in the range [0, 1]
- gt-rating-std: the standard deviation of the normalized numeric ratings.
- gt-rating-median: the median normalized numeric rating in the range [0, 1]
- gt-ratings-raw: a list of strings containing the ratings from each pass of annotation from the set "High", "Moderate", "Low".
- sum-first-section: Wikipedia page summary from the first section of the page
- sum-noun-phrase-spacy: the first 100 noun-phrases from the article
- sum-keywords-spacy: first 100 key phrases using Spacy
- sum-keywords-rake: first 100 key phrases using Rake