Location Aware Modular Biencoder for Tourism Question Answering

Haonan Li[♠,♣] **Martin Tomko**[♡] **Timothy Baldwin**[♠,♣]
♠ School of Computing and Information Systems, The University of Melbourne
♡ Department of Infrastructure Engineering, The University of Melbourne
♣ Department of Natural Language Processing, MBZUAI
haonan.li@mbzuai.ac.ae, tomkom@unimelb.edu.au, tb@ldwin.net

Abstract

Answering real-world tourism questions that seek Point-of-Interest (POI) recommendations is challenging, as it requires both spatial and non-spatial reasoning, over a large candidate pool. The traditional method of encoding each pair of question and POI becomes inefficient when the number of candidates increases, making it infeasible for real-world applications. To overcome this, we propose treating the QA task as a dense vector retrieval problem, where we encode questions and POIs separately and retrieve the most relevant POIs for a question by utilizing embedding space similarity. We use pretrained language models (PLMs) to encode textual information, and train a location encoder to capture spatial information of POIs. Experiments on a real-world tourism QA dataset demonstrate that our approach is effective, efficient, and outperforms previous methods across all metrics. Enabled by the dense retrieval architecture, we further build a global evaluation baseline, expanding the search space by 20 times compared to previous work. We also explore several factors that impact on the model's performance through follow-up experiments. Our code and model are publicly available at https://github.com/haonan-li/LAMB.

1 Introduction

Question answering (QA) models and recommender systems have undergone rapid development in recent years (Seo et al., 2017; Rajpurkar et al., 2016; Kwiatkowski et al., 2019; Lee et al., 2019; Cui et al., 2020; Hamid et al., 2021). However, personalised question answering is still highly challenging and relatively unexplored in the literature. Consider the example question in Figure 1, in the form of a real-world point-of-interest (POI) recommendation question from a travel forum. Answering such questions requires understanding of the question text with possibly explicit (e.g., *in Dublin*) or vague and ambiguous (e.g., *within walking distance of Grafton Street*) spatial constraints, as well as a fast indexing method that supports large-scale reasoning over both spatial and non-spatial (e.g., *fairly priced restaurants*) constraints.

Question: *Hi! My wife and I are in our late thirties and going to be in Dublin on September 28 and 29th. We are staying in the Grafton Street area. Does anybody have any suggestions for some fairly priced restaurants with great food within walking distance of Grafton Street? Also, What about some good pubs with live local music? (I realize it is a Sun and Mon night and may be slow) Any suggestion would be appreciated! Thanks!*
Answer ID: *11_R_4392*
Answer Name: The Porterhouse Central, 45-47 Nassau Street, Dublin.

Figure 1: An example of real-world POI recommendation question from the TourismQA dataset (Contractor et al., 2021b). Colored text represents constraints relevant to recommending POIs.

Recently, there has been increased interest in geospatial QA. Most approaches focus on querying structured knowledge bases, based on translating natural language questions into structured queries, e.g., using SPARQL (Punjani et al., 2018; Li et al., 2021; Hamzei et al., 2022). Separately, Contractor et al. (2021b) introduced the task of answering POI-seeking questions using geospatial metadata and reviews that describe POIs. In later work, they proposed a spatial–textual reasoning network that uses distance-aware question embeddings as input and encodes question–POI pairs using attention (Contractor et al., 2021a). However, as their model creates separate question embeddings for each POI, the inference cost increases linearly in the number of POIs, and the model is incompatible with large

Findings of the Association for Computational Linguistics: IJCNLP-AACL 2023, pages 95–109
November 1–4, 2023.

pre-trained models such as BERT (Devlin et al., 2019) or even medium-sized QA models such as BiDAF (Seo et al., 2017).

In this work, we address the question: can we build a more efficient POI recommendation system which supports the use of advanced pre-trained language models as the textual encoder? By presenting the **L**ocation **a**ware **m**odular **b**i-encoder ("LAMB") model. We use a bi-encoder architecture to encode questions and POIs separately, where the question encoder is a textual module and the POI encoder consists of a textual and a location module. By encoding them separately, we cast the task as a retrieval problem based on dense vector similarity between the question and each POI. For training, we combine each question with one positively-labeled POI and multiple negatively-labeled POIs, and use contrastive learning to train the question encoder and POI encoder simultaneously, by maximizing the similarity between the question and positive POI. After training, we generate location-aware dense representations for all POIs using the POI encoder, and index them by city name and entity (POI) type. For inference, we use the question encoder to generate a location-aware representation, and rank the POIs using similarity.

Our contributions are four-fold: (1) we propose a location-aware modular bi-encoder model which fuses spatial and textual information; (2) we demonstrate that the proposed model outperforms the existing SOTA on a real-world tourism QA dataset, with huge improvements in training and inference efficiency; (3) we build new global evaluation baselines by expanding the search space 20× over local evaluation; and finally, (4) we analyse the influence of different training strategies and hyper-parameters through extensive experiments.

2 Methodology

In this section, we first formulate the task, and then introduce the POI pre-processing method and the LAMB model. Finally, we describe the efficient training and inference strategies.

2.1 Task Formulation

Given a question q, the task is to find the most probable POI answer p from a candidate pool P, which satisfies spatial and non-spatial constraints in q. Each POI in P consists of a *geo-coordinates* $(lat, long)$ of the POI, the multi-granularity location *name* (POI entity name, street, city, postcode), and a list of textual *reviews* $= (r_1, r_2, ...r_n)$. It can be represented as $p = \langle coordinates, name, reviews \rangle$ (see Appendix A for an example).

2.2 POI Pre-processing

Reviews of POIs provide useful information to represent POIs, however, each candidate can have hundreds of reviews, the total length greatly exceeding the maximum token length of 512 tokens in general PLMs such as BERT. To choose more representative reviews, previous work (Contractor et al., 2021b) has clustered reviews into K clusters, and then represented the POI using the top-N sentences from each cluster based on distance from the cluster centroid, resulting in $N \times K$ sentences. However, this approach is potentially problematic as clusters can be of varying size and density, and outliers can affect the centroid. To keep representative reviews, K and N should not be too small, e.g., Contractor et al. (2021b,a) set $N = K = 10$.

In this paper, we adopt the SELSUM (Bražinskas et al., 2021) model, which consists of a selector to choose the M most representative reviews and a summarizer to generate a summary of the selected reviews. We use a model pre-trained on the AMASUM dataset, which includes verdicts, pros, and cons, and hundreds of reviews for more than 31,000 summarized Amazon products (see example in Appendix C). We compare the results using clustering, the selection module only, and the full SELSUM model in Appendix C. Our results show that using a 3-sentence summary for each POI achieves comparable results with a clustering approach that represents each POI via 100 sentences, and that using 10 sentences outperforms the clustering method.

2.3 Location Aware Modular Bi-encoder

LAMB (see Figure 2) uses a bi-encoder framework to encode questions and POIs. The question encoder is a textual module which takes question text as input, and outputs dense representations. The POI encoder consists of a textual module and a location module, where the textual module encodes a description and/or reviews associated with it, and the location module encodes the multi-granularity location names. The outputs of the textual and location modules are real-valued vectors, which are concatenated to represent a POI. Full details of the model are presented below.

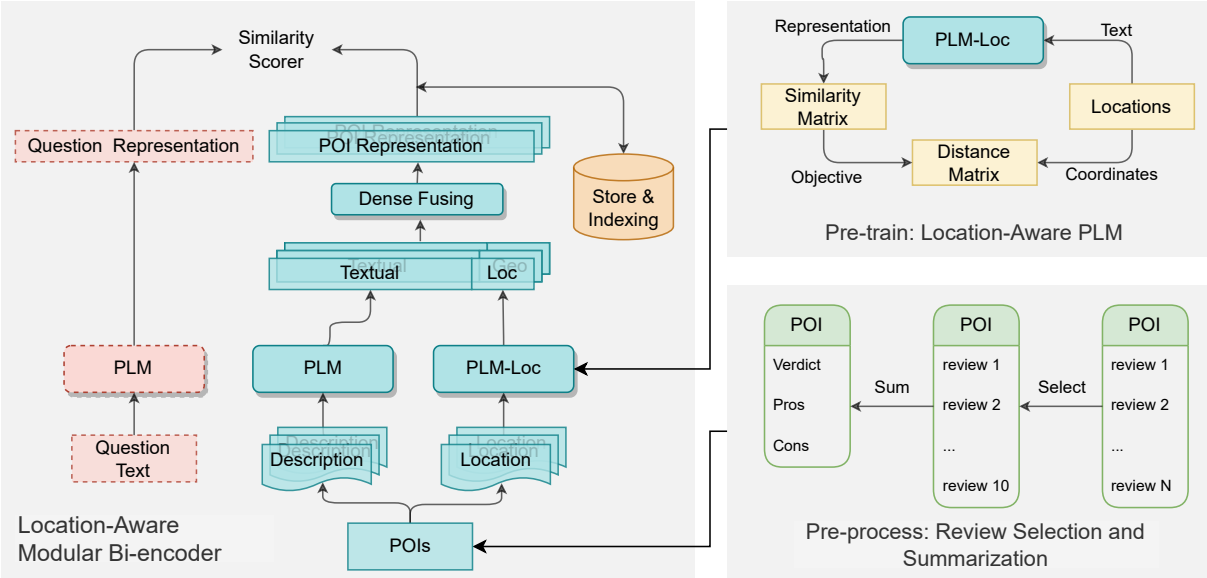

Figure 2: Proposed approach. The reviews of POIs are first selected and summarized by SELSUM (bottom right part). A location module is separately pre-trained under the supervision of geocoordinate-based distances (top right part). The left part is the main LAMB model. The cyan- and salmon-coloured parts are the POI encoder and question encoder, respectively. The orange part is the index of POI embeddings used for inference.

Textual Module We use two independent PLMs as the textual encoder for questions and POIs, using the `[CLS]` token representation as the output. For questions, we do not preprocess the question text, while for POIs, we concatenate the preprocessed reviews.

Location Module Spatial constraints are crucial in retrieving relevant POIs to a question. However, previous research has shown that PLMs perform poorly in encoding and reasoning over spatial data, especially for geolocation information (Scherrer and Ljubešić, 2021; Hofmann et al., 2022). To enhance the model's ability to capture geospatial information, we employ a location module that explicitly encodes the multi-granularity location name of a POI into a dense vector. We initialize the location module by choosing several transformer blocks from a PLM, and continue pre-training it to learn geo-coordinate-aware location name representations. The training object is designed to pull together pairs of encoded location representations if the locations are physically near each other, and push them apart if they are far from each other.

Formally, for any three POIs (p_0, p_1, p_2), suppose the corresponding locations are (l_0, l_1, l_2), and the encoded representations are (h_0, h_1, h_2). Here $l_i (i = 0, 1, 2)$ is a 1-d vector $[lat_i, long_i]$, representing the latitude and longitude of p_i, with $lat_i \in [-90, 90]$ and $long_i \in [-180, 180]$, and h_i is a vector. We choose p_0 to be an anchor location, and $d_i (i = 1, 2) \in [0, 1]$ to represent the normalized Haversine distance between l_0 and l_i, representing the greater-circle distance between two points on a sphere. Similarly, $s_i (i = 1, 2) \in [0, 1]$ represents the cosine similarity between h_0 and h_i. We use the triplet margin loss, and define the loss function as follows:

$$\mathcal{L} = \begin{cases} \max((s_1 - s_2) + (d_1 - d_2), 0) & \text{if} (d_1 - d_2) > 0 \\ \max((s_2 - s_1) - (d_1 - d_2), 0) & \text{otherwise} \end{cases}$$

In the first case, $d_1 - d_2 > 0$ means that p_2 is closer to p_0 than p_1, and hence we structure the loss to learn a larger s_2 (= higher similarity between p_0 and p_2) and smaller s_1 (= lower similarity between p_0 and p_1). We set the difference between the two distances as a dynamic margin, which controls the rationally-valued similarity difference.

Question and POI Encoders As mentioned above, we use a separate textual encoding module E_P^{text} and location encoding module E_P^{loc} to encode each POI. These modules map the review text and location names to fixed-length vectors:

$$r_p^{text} = E_P^{text}(p) \in \mathbb{R}^{1 \times d_1}$$
$$r_p^{loc} = E_P^{loc}(p) \in \mathbb{R}^{1 \times d_2}$$

We concatenate r_p^{text} and r_p^{loc} and then use a dense layer to fuse the representations together, resulting

in the POI representation $r_p \in \mathbb{R}^{1\times d}$:

$$r_p = \text{Dense}([r_p^{text}, r_p^{loc}]) \in \mathbb{R}^{1\times d}$$

For questions, we similarly tried using separate text and location modules, and combining their outputs. However, we found that the text may contain distractor locations that should not be considered as spatial constraints, and that context is essential. (e.g., the place name *Italy* in question *Hey I am from Italy, please suggest a restaurant in Berlin that suits my appetite.*) Hence, we use a single textual module E_Q^{text} which directly maps the question text into representation $r_q \in \mathbb{R}^{1\times d}$, of the same dimension as a POI.

2.4 Training and Inference

We train the two encoders simultaneously using contrastive learning. We input each question q_i with one positive POI p_i^+ and several negative POIs $p_{i,1}^-, ...p_{i,n}^-$ into the model, with the objective to maximize the similarity between the embeddings of q_i and p_i^+, while minimizing the similarity between the embeddings of q_i and $p_{i,1}^-, ...p_{i,n}^-$. We use the negative log-likelihood (NLL) loss of the positive POIs as our objective function:

$$\begin{aligned}&\mathcal{L}(q_i, p_i^+, p_{i,1}^-, ...p_{i,n}^-)\\ &= -\log \frac{e^{\text{sim}(q_i,p_i^+)}}{e^{\text{sim}(q_i,p_i^+)} + \sum_{j=1}^{n} e^{\text{sim}(q_i,p_{i,j}^-)}}\end{aligned}$$

where similarity function $\text{sim}(p, q)$ is the inner product.

Negative Sampling Strategy A critical question in contrastive learning is how to construct positive and negative examples. In our case, for each question, there can be more than one answer (= positive) POI. To make use of every positive POI, as well as to adapt to the NLL loss function, we create a training example for each positive POI. For negative samples, all non-answer POIs are candidate negative samples, but previous work (Karpukhin et al., 2020; Xiong et al., 2021a) has shown that high-quality negative samples help to learn a better encoder. In this research, we consider three different types of negative samples: (1) *easy negatives* = random (non-answer) POIs from the entire candidate set; (2) *medium negatives* = random (non-answer) POIs that are in the same city and of the same type (restaurant, attraction, or hotel) as the answer POI; and (3) *hard negatives* = top-k ranked non-answer POIs from the previous epoch.

Two-phase Training We conduct two phases of training: first, we use easy and medium negatives to do warm-up training of the model, and provide the model with a relatively easily-optimizable objective; next, we switch over to training with a mixture of medium and hard negatives.[1] We sample hard negatives by performing inference on the training data after each epoch (or a specific number of steps) to find the top-k POIs for each training question. We then create new training instances by randomly sampling N non-answer POIs from the top-k retrieved POIs, and use these to continue training the model.

Inference Before inference, we disable the question encoder and generate representations of all POIs using the POI encoder only, and store and index them (as shown in the orange part in Figure 2). During inference, the generated POI representations are loaded into memory. Given a question q at run-time, we encode it using the question encoder, score all candidates using the pre-computed representations, and return the top-k results.

3 Experimental Setup

In this section, we introduce the dataset, baselines, and implementation details of our model.

3.1 Dataset

We use the TourismQA (Contractor et al., 2021b) dataset, which comprises over 47,000 real-world POI question–answer pairs from 50 cities across the globe. These questions are genuine queries submitted to a trip advisor website,[2] and the answers are real-world responses that have been chosen and authenticated by annotators. The average length of the questions is 87.48 tokens (separated by whitespace). And on average, there are 3.63 POIs as ground truth answers for each question. The dataset contains roughly 114,000 candidate POIs altogether, each with a collection of reviews and metadata such as geo-coordinates and type (restaurant, attraction, or hotel).

We follow Contractor et al. (2021b) in dividing the dataset into a 9:1 train–test split, and constructing a search space by including POIs located in the same city as the ground truth POIs, resulting in an average of approximately 5,300 candidate POIs per question. We believe one reason for earlier work

[1] For convenience, we use "easy" and "hard" negatives to describe the training setting in any single phase.

[2] https://www.tripadvisor.in

to build the candidate pool within a city was that their methods struggled with a large candidate pool. However, in real-world scenarios, the ground truth answer is concealed, and the candidate pool may be extensive, encompassing all POIs in the database. Therefore, we established a new evaluation setting in which the search space comprises all POIs in the world. We refer to this new setting as *global* evaluation (114,000 candidates), and the previous one as *local* evaluation (5,300 candidates).

3.2 Evaluation Metrics

Following Contractor et al. (2021b), we evaluate using Accuracy@$N \in \{3, 5, 30\}$ and mean reciprocal rank (MRR) for local evaluation, and use Accuracy@$N \in \{5, 30, 100\}$ for global evaluation. For Accuracy@N, if the top-N predictions have a non-empty intersection with the answer POI set, the results are considered to be correct. For MRR, we return the reciprocal rank of the first positive answer POI per question, and average over the questions.

3.3 Baselines

We compare ourselves against four baselines, as detailed below.

Sort by Distance (SD): Given all tagged locations with geo-coordinates in the question, we rank POIs by the minimal distance from the tagged locations.

BM25: We represent each POI by its combined reviews, and index them using Apache Lucene. Then questions are used as a query to compute BM25 scores for all POIs.

Cluster-Select-Rerank ("CSR") Model (Contractor et al., 2021b), which consists of three components: (1) a clustering module that clusters reviews for each POI and selects representative reviews; (2) a Duet (Mitra and Craswell, 2019) retrieval model that selects the best 30 candidate POIs; and (3) a QA-style re-ranker that scores and re-ranks the selected POIs. Note that the cluster module is used to pre-process the POIs, and the selection and re-ranking modules are trained separately and pipelined.

Spatial-Textual CSR (Contractor et al., 2021a), which adds a self-attention based geospatial reasoner to the CSR model, and ranks POIs based on the weighted sum of scores from the geo-spatial reasoner and CSR.

3.4 LAMB Implementation Details

We implement our model in PyTorch, and use the HuggingFace (Wolf et al., 2020) implementation of DistilBERT (Sanh et al., 2019) as the textual encoder. The location module is comprised of two transformer blocks that are initialized using the first two blocks of a pre-trained DistilBERT model. We continued pre-training for 3 epochs using triplet loss to force the model to learn more spatial information, as described in Section 2.3. During this process, we set the batch size to 8, learning rate to 2e-5, and the max sequence length to 64.

For the main model of LAMB, the maximum length (in subtokens) for both questions and reviews is set to 256. For training, we use a linear learning rate scheduler with an initial learning rate of 2e-5, and the Adam optimizer with default hyperparameters. For each training instance, we use a single positive POI and varying numbers of negatives. We set the batch size to 8 and train for 10 epochs: 5 epochs of phase 1 (easy and medium negatives), and 5 epochs of phase 2 (medium and hard negatives). All experiments were run on a single Nvidia A100 40GB GPU for about 8 hours.

4 Results and Analysis

Table 1 shows the overall performance of the baselines and our proposed model. We can see that the sparse-vector retrieval (BM25) and distance-based retrieval (SD) models in the first block of the table perform extremely poorly, demonstrating the difficulty of the task. In contrast, the textual-only pipelined models (CRQA and CSRQA) in the second block improve overall performance substantially, and adding the spatial reasoning subnetwork ("ST+") boosts results again. Note that, since CSRQA is pipelined with a selection model that selects the top-30 results, the spatial-textual module cannot improve Accuracy@30 further.

Compared to the baselines in blocks one and two, our model, LAMB, achieves the state-of-the-art across all metrics. To better understand the impact of different components of our model, we conducted an ablation study by separately removing the training phase 2, review selection and summarization modules, and location module. Overall, the performance dropped when one of these modules or strategies was removed, but still outperformed the previous state-of-the-art. Specifically, removing training phase 2 had a relatively large impact on local evaluation, which we attribute to the process

Model	Local				Global			
	Acc@3	Acc@5	Acc@30	MRR	Acc@5	Acc@30	Acc@100	MRR
SD	0.83	1.11	5.62	0.011	0.83	1.11	5.62	0.011
BM25	5.59	8.29	16.33	0.061	0.44	1.86	3.68	0.014
CRQA	16.89	23.75	52.51	0.159	–	–	–	–
ST+CRQA	19.37	26.23	56.33	0.175	–	–	–	–
CSRQA	21.44	28.20	52.65	0.186	–	–	–	–
ST+CSRQA	22.41	28.99	52.65	0.193	–	–	–	–
LAMB	**24.83**	**32.51**	**60.92**	**0.220**	**14.07**	**32.87**	49.08	**0.101**
−Phase 2	22.49	29.35	59.20	0.201	13.22	31.89	**49.68**	0.094
−SELSUM	23.90	31.20	60.52	0.216	13.28	32.12	48.63	0.096
$-E_{loc}$	23.68	31.30	60.52	0.215	9.59	24.52	40.03	0.071

Table 1: Overall evaluation on the TourismQA dataset. The second block of results are based on the TourismQA paper, wherein the best results are underlined, and "ST" denotes the spatial–textual module. The overall best results are in **bold**. The third block presents the results for the full LAMB model, and also with module ablation.

Model	Training	Inference	
	Time (h)	#Cand	Time (h)
CRQA (±ST)	360	5.3k	64
CSRQA (±ST)	360+	30	2–3
LAMB	10	115k	0.15

Table 2: Runtime comparison, based on a single Nvidia V100 GPU. "#Cand" indicates the number of candidate POIs. For CSRQA, time was estimated by summing the times of the component models.

Loc Module	Acc@5	Acc@30	Acc@100	MRR
w/o Loc	9.59	24.52	40.03	0.071
2-l PLM	9.91	25.47	41.64	0.075
1-l PLM-Loc	12.86	30.67	46.28	0.091
2-l PLM-Loc	**14.07**	**32.87**	**49.08**	**0.101**
4-l PLM-Loc	10.92	27.78	43.24	0.081

Table 3: Results with different location module settings. "PLM" = use PLM directly; "PLM-Loc" = continue to pretrain PLM on location names; and "N-l" = use N transformer blocks.

of training to distinguish hard negatives. Removing the location module greatly impacted the global evaluation, demonstrating the effectiveness of the location module, particularly when candidates are from around the globe.

Based on our analysis, there are three main reasons why LAMB outperforms previous models: (1) training and inference are end-to-end, avoiding error propagation due to pipelining, as with CSRQA; (2) our use of pre-trained language models as the textual encoder, outperforming static word embeddings or training encoders from scratch; and (3) learning location encodings separately and fusing them with textual representations, providing a soft distance computing method. We provide a comparison between our location module design and other straightforward geo-coordinate-based location/distance modules in Appendix E. From this, we can conclude that compared to strategies that encode geo-coordinates directly, a pretrained location name module better captures spatial information.

4.1 Efficiency Comparison

We analyze the computational requirements of the models in Table 2. LAMB is more time efficient than the previously-proposed neural models, requiring around 5% of the training time, and <10% of the inference time. It is also able to handle a much larger candidate pool (in the millions of candidates) compared to C(±S)RQA (in the tens or thousands of candidates). Further analysis of efficiency and usability is provided in Appendix D.

4.2 Ablation Study on Model Training

To further understand how different model training options affect the results, we conduct several additional experiments and discuss our findings below.

Location Module Analysis In this section, we compare various settings of location modules as shown in Table 3. The table indicates that continuous pretraining of a PLM on location names significantly enhances the module's ability to capture geo-location and distance. Furthermore, using two transformer blocks is sufficient to encode multi-granularity location names, whereas more or fewer layers may lead to overfitting or underfitting.

Effectiveness of Negative Examples To investigate the effectiveness of the type and number of negative examples during training, we kept the total

#HN	Local				Global			
	Acc@3	Acc@5	Acc@30	MRR	Acc@5	Acc@30	Acc@100	MRR
0	16.41	22.59	51.05	0.159	19.51	**44.70**	**64.91**	**0.137**
1	15.51	21.34	51.67	0.154	16.97	43.06	62.19	0.123
4	20.55	27.29	52.41	0.188	**20.13**	40.43	56.27	0.142
8	23.99	30.58	59.47	0.213	17.37	37.78	54.20	0.124
12	**24.83**	**32.51**	**60.92**	**0.220**	14.07	32.87	49.08	0.101
15	24.06	31.44	60.70	0.221	8.10	21.34	36.18	0.063

Table 4: Results with differing numbers of easy/hard negatives, total negatives = 15. #HN: number of hard negatives.

#Phase 1,2	Local				Global			
	Acc@3	Acc@5	Acc@30	MRR	Acc@5	Acc@30	Acc@100	MRR
10, 0	22.49	29.35	59.20	0.201	13.22	31.89	49.68	0.094
8, 2	23.08	30.22	60.33	0.209	13.48	32.48	**50.65**	0.096
5, 5	**24.83**	**32.51**	**60.92**	**0.220**	**14.07**	**32.87**	49.08	**0.101**
2, 8	24.49	31.88	60.08	0.219	13.05	31.18	46.04	0.095
0, 10	21.73	28.28	54.15	0.198	11.70	28.74	42.82	0.085

Table 5: Results with varied epochs in two-phase training, using 10 total training epochs.

number of negatives constant at 15 while varying the mix of easy and hard negatives (as presented in Table 4). As we increase the number of hard negatives, the global evaluation results deteriorate while the local evaluation results improve. This implies that training with easy negatives is more appropriate when the target city or area is unconstrained. The best local evaluation results were achieved when using 12/15 hard negatives, indicating that easy negatives are still necessary for learning general location constraints. We further investigated varying the total number of negatives for contrastive learning, as presented in Table 7 in the Appendix. Our findings indicate that the more negatives we have in each training instance, the better the model performs, but that the relative improvement plateaus beyond around 30.

Two-Phase Training Strategy We conducted experiments with different epoch configurations for our two-phase training strategy, as detailed in Table 5. Our results indicate that both phase 1 and phase 2 are essential, aligning with the assumptions stated in Section 2.4. Furthermore, we found that commencing phase 2 training at the midway point was particularly effective.

4.3 Human Evaluation

To further investigate the dataset and have a better sense of the overall performance of LAMB, we conducted a small-scale human evaluation. We randomly choose 100 questions from the test set and manually evaluate the top-3 predictions for relevance based on LAMB as presented in Table 1. For this small question set, our estimate of the true Accuracy@3 is around 75%, as compared to the automatic evaluation result of 24%. This is consistent with the human evaluation results reported in (Contractor et al., 2021a), and points to the issue of low label-recall in the dataset: while a given POI may not have been selected by the user who issued the original question, it may well have satisfied the constraints described in the question.

4.4 How ChatGPT Performs on TourismQA

During the writing of this paper, ChatGPT (i.e. GPT3.5) was released. We manually tested 100 questions from Section 4.3 by inputting them directly into ChatGPT (GPT-3.5-turbo on 20-March-2023) and getting a single response.[3] The results show that out of the 100 questions, 91 received recommendations for points of interest or areas. However, only 14 of those replies match the ground truth answers, which is lower than our model's performance of 24. We believe that the main reason for this discrepancy is due to differences in the POI databases. The replies from ChatGPT were well-organized and logical, and could even answer many details in the questions beyond the capabilities of our model.

However, we observed that ChatGPT failed to provide an output in many cases: among the 100 replies, sentences such as *As an AI language model,*

[3]Questions and responses are released together with the source code.

I don't have personal experience in ... appeared 36 times, while other outputs like *I can recommend that you check out the reviews on websites like TripAdvisor or Booking.com* appeared 13 times. Additionally, ChatGPT tended to recommend popular places, with the word *popular* appearing 44 times in replies, despite not being mentioned in any of the questions. We observed further bias in ChatGPT's recommendations. For example, it recommended *Shake Shack* nine times in response to fast food requests, but never mentioned other international fast-food chains or local chains, even when questions specifically asked for fast food with regional characteristics.

Lastly, ChatGPT's database is not up-to-date, as also mentioned in its replies. Since OpenAI did not provide full training details, the cost of updating the database, including fine-tuning the model, is unclear. In summary, there is still a real need for a comprehensive recommendation system that can be combined with up-to-date website information.

5 Related Work

Geo-Spatial Question Anwering There has been a strong focus in the literature on component geospatial tasks such as geo-parsing (toponym recognition and disambiguation) (Karimzadeh et al., 2019; Wang et al., 2020a), geo-tagging (tagging toponyms with geographic metadata) (Compton et al., 2014; Middleton et al., 2018), geospatial information retrieval (Purves et al., 2018), and geospatial question analysis (Hamzei et al., 2019).

Based on the type of question, existing work on geospatial QA ("GeoQA") can be classified into four types (Mai et al., 2021): (1) factoid GQA (Li et al., 2021; Hamzei et al., 2022), focusing on answering questions with geographic factoids; (2) geo-analytical QA (Scheider et al., 2020; Xu et al., 2020), focusing on questions with complex spatial analytical intent; (3) visual GQA (Lobry et al., 2020; Janowicz et al., 2020), linking questions to an image or video; and (4) scenario-based GQA (Huang et al., 2019; Contractor et al., 2021b), which associates questions with a scenario described with a map or paragraph of text. Our work corresponds to the last type, and unlike most other work, we do not rely on task-specific query languages or annotations, and focus more on NLP and IR modeling.

Point-of-Interest (POI) Recommendation POI recommendation systems have a wide range of applications such as online navigation applications (Zhao et al., 2019a; Yuan et al., 2021), personalized recommendation systems in location-based social networks (Feng et al., 2015; Zhao et al., 2019b), and trip or accommodation advisory systems (Li et al., 2016; Contractor et al., 2021b). In this research, we focus on POI recommendation incorporating both structured information (such as geo-coordinates) and unstructured information (such as textual descriptions). Previous work has explored efficient spatial indexing based on specialized data structures, with textual information as sparse vectors or filters (de Almeida and Rocha-Junior, 2015; Li et al., 2016). Recent work (Contractor et al., 2021b,a) has focused on latent textual representations, which is highly relevant here.

Textual Encoding and Document Retrieval Pretrained language models (PLMs) have led to great successes across many NLP tasks (Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019; Lewis et al., 2020; He et al., 2021; Clark et al., 2020). In the field of QA, PLMs have been used to generate representations of questions and documents (Nogueira et al., 2019; Zhang et al., 2020). In this work, we use DistilBERT (Sanh et al., 2019) as our textual encoder, as it is more efficient than BERT and retains much of its expressivity.

Document retrieval has become a mainstay of research in IR and QA. Recently, IR has increasingly moved towards dense vector retrieval methods (Das et al., 2019; Seo et al., 2019; Xiong et al., 2021b). In particular, Karpukhin et al. (2020) proposed DPR based on a dual-encoder approach, and attained impressive results on multiple open-domain question answering benchmarks. Inspired by this, we adopt a bi-encoder framework.

6 Conclusion

We have proposed the LAMB model, a location-aware bi-encoder model for answering POI recommendation questions. Experiments on a recently-released tourism question-answering dataset show that our model surpasses existing spatial-textual reasoning models across all metrics. Experiments over LAMB's components and based on changing up the training strategy show the effectiveness of the different design choices used in LAMB. Finally, we analyzed the training and inference efficiency, and demonstrated that our model is resource-efficient at training and inference time, suggesting it can be deployed in real-world tourism

applications.

Limitations

Although we have achieved results that significantly outperform the current state-of-the-art, our work still has some limitations. First, as demonstrated in Section 4.3 and in the earlier work of Contractor et al. (2021a), the TourismQA dataset was collected semi-automatically, and the gold labels have high precision but low recall. Hence any results on this dataset are likely an underestimate of the true model performance. While we currently use the Haversine formula to compute the distance between two locations and supervise the pre-training of the location module, we recognize that this calculation may not reflect the actual distance between two places, taking into account the route direction and vertical height difference. In light of the city's urban design, the Manhattan distance might better represent the true distance between two locations within a city. Additionally, POI density could be a factor that influences user choice in real life, in that people may be more inclined to go to locations with a higher density of restaurants to eat (in order to have more options if a given restaurant doesn't live up to their expectations), rather than travel far to a remote place without other options in the local vicinity. For hotels, on the other hand, some users may prefer privacy and a lower density. Such extra-linguistic features are not explicitly captured in our model.

References

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2021. Learning opinion summarizers by selecting informative reviews. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *Proceedings of the 8th International Conference on Learning Representations*. OpenReview.net.

Ryan Compton, David Jurgens, and David Allen. 2014. Geotagging one hundred million twitter accounts with total variation minimization. In *Proceedings of the 2014 IEEE International Conference on Big Data*, pages 393–401. IEEE Computer Society.

Danish Contractor, Shashank Goel, and Parag Singla. 2021a. Joint spatio-textual reasoning for answering tourism questions. In *Proceedings of the Web Conference 2021*, pages 1978–1989.

Danish Contractor, Krunal Shah, Aditi Partap, Parag Singla, and Mausam. 2021b. Answering poi-recommendation questions using tourism reviews. In *the 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 281–291. ACM.

Zhihua Cui, Xianghua Xu, Xue Fei, Xingjuan Cai, Yang Cao, Wensheng Zhang, and Jinjun Chen. 2020. Personalized recommendation system based on collaborative filtering for IoT scenarios. *IEEE Transactions on Services Computing*, 13(4):685–695.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019. Multi-step retriever-reader interaction for scalable open-domain question answering. In *Proceedings of the 7th International Conference on Learning Representations*. OpenReview.net.

João Paulo Dias de Almeida and João B. Rocha-Junior. 2015. Top-k spatial keyword preference query. *Journal of Information and Data Management*, 6(3):162–177.

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

Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized ranking metric embedding for next new POI recommendation. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 2069–2075. AAAI Press.

Rula A Hamid, Ahmed Shihab Albahri, Jwan K Alwan, ZT Al-Qaysi, Osamah Shihab Albahri, AA Zaidan, Alhamzah Alnoor, AH Alamoodi, and BB Zaidan. 2021. How smart is e-tourism? a systematic review of smart tourism recommendation system applying data management. *Computer Science Review*, 39:100337.

Ehsan Hamzei, Haonan Li, Maria Vasardani, Timothy Baldwin, Stephan Winter, and Martin Tomko. 2019. Place questions and human-generated answers: A data analysis approach. In *Proceedings of the 22nd AGILE Conference on Geographic Information Science*, pages 3–19. Springer.

Ehsan Hamzei, Martin Tomko, and Stephan Winter. 2022. Translating place-related questions to geosparql queries. In *Proceedings of the ACM Web Conference 2022*, pages 902–911. ACM.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: decoding-enhanced Bert with disentangled attention. In *Proceedings of the 9th International Conference on Learning Representations*. OpenReview.net.

Valentin Hofmann, Goran Glavaš, Nikola Ljubešić, Janet B Pierrehumbert, and Hinrich Schütze. 2022. Geographic adaptation of pretrained language models. *arXiv preprint arXiv:2203.08565*.

Zixian Huang, Yulin Shen, Xiao Li, Yu'ang Wei, Gong Cheng, Lin Zhou, Xinyu Dai, and Yuzhong Qu. 2019. GeoSQA: A benchmark for scenario-based question answering in the geography domain at high school level. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5866–5871, Hong Kong, China. Association for Computational Linguistics.

Krzysztof Janowicz, Song Gao, Grant McKenzie, Yingjie Hu, and Budhendra Bhaduri. 2020. GeoAI: Spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond. *International Journal of Geographic Information Science*, pages 625–636.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with GPUs. *IEEE Trans. Big Data*, 7(3):535–547.

Morteza Karimzadeh, Scott Pezanowski, Alan M MacEachren, and Jan O Wallgrün. 2019. Geotxt: A scalable geoparsing system for unstructured text geolocation. *Transactions in GIS*, 23(1):118–136.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. 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.

Haonan Li, Ehsan Hamzei, Ivan Majic, Hua Hua, Jochen Renz, Martin Tomko, Maria Vasardani, Stephan Winter, and Timothy Baldwin. 2021. Neural factoid geospatial question answering. *Journal of Spatial Information Science*, 23:65–90.

Miao Li, Lisi Chen, Gao Cong, Yu Gu, and Ge Yu. 2016. Efficient processing of location-aware group preference queries. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*, pages 559–568. ACM.

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. *ArXiv preprint*, abs/1907.11692.

Sylvain Lobry, Diego Marcos, Jesse Murray, and Devis Tuia. 2020. RSVQA: Visual question answering for remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 58(12):8555–8566.

Gengchen Mai, Krzysztof Janowicz, Rui Zhu, Ling Cai, and Ni Lao. 2021. Geographic question answering: Challenges, uniqueness, classification, and future directions. *AGILE: GIScience Series*, 2:1–21.

Stuart E Middleton, Giorgos Kordopatis-Zilos, Symeon Papadopoulos, and Yiannis Kompatsiaris. 2018. Location extraction from social media: Geoparsing, location disambiguation, and geotagging. *ACM Transactions on Information Systems (TOIS)*, 36(4):1–27.

Bhaskar Mitra and Nick Craswell. 2019. An updated duet model for passage re-ranking. *ArXiv preprint*, abs/1903.07666.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *ArXiv preprint*, abs/1904.08375.

Dharmen Punjani, K Singh, Andreas Both, Manolis Koubarakis, Ioannis Angelidis, Konstantina Bereta, Themis Beris, Dimitris Bilidas, T Ioannidis, Nikolaos Karalis, and C. Lange. 2018. Template-based question answering over linked geospatial data. In *Proceedings of the 12th Workshop on Geographic Information Retrieval*, page 7.

Ross S Purves, Paul Clough, Christopher B Jones, Mark H Hall, and Vanessa Murdock. 2018. Geographic information retrieval: Progress and challenges in spatial search of text. *Foundations and Trends in Information Retrieval*, 12(2-3):164–318.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv preprint*, abs/1910.01108.

Simon Scheider, Enkhbold Nyamsuren, Han Kruiger, and Haiqi Xu. 2020. Geo-analytical question-answering with GIS. *International Journal of Digital Earth*, pages 1–14.

Yves Scherrer and Nikola Ljubešić. 2021. Social media variety geolocation with GeoBERT. In *Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects*. The Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *Proceedings of the 5th International Conference on Learning Representations*. OpenReview.net.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4430–4441, Florence, Italy. Association for Computational Linguistics.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Jimin Wang, Yingjie Hu, and Kenneth Joseph. 2020a. NeuroTPR: A neuro-net toponym recognition model for extracting locations from social media messages. *Transactions in GIS*, 24(3):719–735.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020b. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021a. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the 9th International Conference on Learning Representations*. OpenReview.net.

Wenhan Xiong, Hong Wang, and William Yang Wang. 2021b. Progressively pretrained dense corpus index for open-domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2803–2815, Online. Association for Computational Linguistics.

H. Xu, E. Hamzei, E. Nyamsuren, H. Kruiger, S. Winter, M. Tomko, and S. Scheider. 2020. Extracting interrogative intents and concepts from geo-analytic questions. *AGILE: GIScience Series*, 1:23.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pages 5754–5764.

Zixuan Yuan, Hao Liu, Junming Liu, Yanchi Liu, Yang Yang, Renjun Hu, and Hui Xiong. 2021. Incremental spatio-temporal graph learning for online query-poi matching. In *Proceedings of the Web Conference 2021*, pages 1586–1597. ACM / IW3C2.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020. SG-Net: Syntax-guided machine reading comprehension. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 9636–9643. AAAI Press.

Ji Zhao, Dan Peng, Chuhan Wu, Huan Chen, Meiyu Yu, Wanji Zheng, Li Ma, Hua Chai, Jieping Ye, and Xiaohu Qie. 2019a. Incorporating semantic similarity with geographic correlation for query-poi relevance learning. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pages 1270–1277. AAAI Press.

Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S. Sheng, and Xiaofang Zhou. 2019b. Where to go next: A spatio-temporal gated network for next POI recommendation. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pages 5877–5884. AAAI Press.

A POI Example

Table 6 shows a POI example, from which we can see that many reviews have similar semantics, making it important to choose representative reviews. In this work, we cluster sentences from reviews, and choose reviews evenly from each cluster to make up the textual input.

B Impact of Total Number of Negative Examples

Table 7 presents experimental results with differing numbers of total negative examples. As the training process is based on contrastive learning, increasing the number of negative examples within a batch leads to an improvement in the model's performance.

C SELSUM Example and Effectiveness

Figure 3 shows an example of SELSUM model output. Table 8 presents the comparison of using clustered reivews, selected reviews (of SELSUM), and summarized reviews.

D Efficiency and Usability Analysis

The most important component of LAMB is the textual encoder, which can be replaced by any pre-trained language model. With the increased development of model distillation and compression methods (Jiao et al., 2019; Wang et al., 2020b; Sun et al., 2020), LAMB can be made more space and time efficient with advanced encoders. Here, we analyse the model's efficiency and usability by considering: (1) adding a new POI into the database; (2) answering a new question; and (3) maintaining the high accuracy of the model.

New POI: To add a new POI to the candidate set, the first step is to do inference using SELSUM model. It is then fed into the POI encoder, and stored for inference purposes. The primary costs of GPU training time and GPU memory consumption can be ignored.

New Question: Given a new question, we input it into the question encoder without any preprocessing such as geo-parsing or tagging. After encoding, LAMB ranks the candidate POIs according to vector similarity, based on simple vector dot product. In this paper, we didn't use any special techniques to speed this up, but in practical applications, techniques such as FAISS (Johnson et al., 2021) can be used to achieve sub-linear times.[4]

Training and Update: The training of LAMB takes no more than 12 hours on a single GPU. Figure 4 shows the top-k retrieval accuracy with respect to the number of training epochs, based on which we can see that the model already achieves good results after 5 epochs. Once this has happened, there is no need to retrain the model from scratch: as more and more new questions and POIs appear, to maintain high performance of the model, it should be enough to fine-tune it on the new questions and POIs for one or two additional epochs.

E Comparison to Geo-coordinate-based Location/Distance Module

We compare our location module with straightforward geo-coordinate-based location and distance modules. Specifically, during question preprocessing, we detect location mentions and tag them with geo-coordinates using a geo-tagger. Similar to LAMB, the question location module E_Q^{loc} maps the geo-coordinates of the mentioned locations into fixed-length vectors:

$$r_q^{loc} = E_Q^{loc}([l_1, l_2, ..., l_m]) \in \mathbb{R}^{1 \times d_2}$$

where m is a hyper-parameter determined based on the average number of location mentions in questions ($m = 5$ here). Each l_i is a 2-d vector $[lat_i, long_i]$. If a question contains $n > m$ unique locations, we randomly select m locations as the input to E_Q^{loc}, otherwise we pad the input to m with $[0, 0]$. Note that the output dimension d_2 is fixed and independent of the number of locations n. For POI, we simply set $m = 1$.

Location Module The location modules for both questions and POIs are implemented with a multi-layer perceptron. Since multiple location mentions (geo-coordinates) may exist in a given question while each POI has a unique geolocation, the sizes of the two location modules are slightly different: POIs are represented as $[lat, long]$ (with size = 2), while questions are represented as $[lat_1, long_1, lat_2, long_2, ..., lat_m, long_m]$ (size = $2m$). We use a 3-layer MLP with dropout of 0.2 and ReLU activation function to map locations into a $2m$-d vector (i.e., $d_2 = 2m$).

Distance Module Since the location module indiscriminately encodes location mentions from the

[4] FAISS is an efficient open-source library for approximate nearest-neighbor search.

Key	Value
Name	Donnybrook, 35 Clinton St, New York City, NY 10002-2426
Lat Long	[40.7201861, -73.9846227]
Reviews	The place was far from packed, but those who were there were very loud. It was the only place that we didnt have on a list of places to go and I have to say it was one of the high lights of the night. We stumbled across this place whilst enjoying the night life around the lower east side late on a Saturday night. I went in for drinks while I waited for a reservation nearby. On the whole , the atmosphere was not one in which I'd like to stay very long. The loud music wasn't the problem. Turns out that the place is very noisy. They were not serving food when we arrived, but the bar tender ordered a pizza for us which we ate at the bar :-) Definitely include it in an East Village pub crawl Nice bar to grab a beer and a warm pretzel. I wanted a nice pub to sit down and have a beer in peace and quiet. You always find a seat in this place . A great prerequisite to the delancey for the final blow out. A disappointment, but it depends what you're after. We were early for our res at Ivan Ramen down the street. Good beer and good service . They have Magners (just what you need on a hot NYC summer day) The pace is easy going, the staff are friendly and the drinks are reasonably priced (similar to Dublin). Staff is kindle and guinness is a real guinness . Good draft selection and very friendly service The bartender was very friendly. I can't speak to the food, but it was exactly what we needed when we needed it. I had read the reviews, and had high expectations. For a very young audience, I guess it might be fun The music was R&B / HIPHOP / Pop and the whole place had a really good vibe. Everyone up dancing, lots of new yorkers. We stumbled across this "Irish" bar while waiting to check into our hotel.

Table 6: A POI example, where reviews have been segmented into sentences.

Negatives		Local				Global			
#N	#HN	Acc@3	Acc@5	Acc@30	MRR	Acc@5	Acc@30	Acc@100	MRR
1*	0	14.98	20.75	48.30	0.140	3.57	10.64	21.80	0.020
3*	2	17.22	23.92	52.95	0.164	9.71	28.22	45.03	0.067
7	5	19.76	25.66	56.58	0.182	11.81	29.40	48.81	0.090
31	16	24.24	31.56	**61.33**	0.218	14.51	34.06	50.72	0.105
47	24	**24.57**	**32.05**	60.75	**0.221**	**16.16**	**37.76**	**54.78**	**0.117**

Table 7: Results with differing numbers of total negatives, with around 3/4 hard negatives. Lines with * signify results with early stopping, because using only hard negatives collapsed the model.

Review Module	Local				Global			
	Acc@3	Acc@5	Acc@30	MRR	Acc@5	Acc@30	Acc@100	MRR
Cluster	23.90	31.20	60.52	0.216	13.28	32.12	48.63	0.096
SEL	24.87	32.08	61.17	0.221	13.28	32.05	47.96	0.095
SELSUM	24.83	32.51	60.92	0.220	14.07	32.87	49.08	0.101

Table 8: Comparison of using clustered reviews, selected reviews with SELSUM, and summarized reviews with SELSUM.

question into a fixed-length vector, some of which may be irrelevant or even harmful for POI matching, we add a distance module to explicitly compute a distance score from the location mentions in the question to a POI, followed by min-pooling to choose the minimal distance from the question to a given POI. We use the Haversine formula to compute distances.

To use the distance module, we define similarity between a question q and a POI p using the

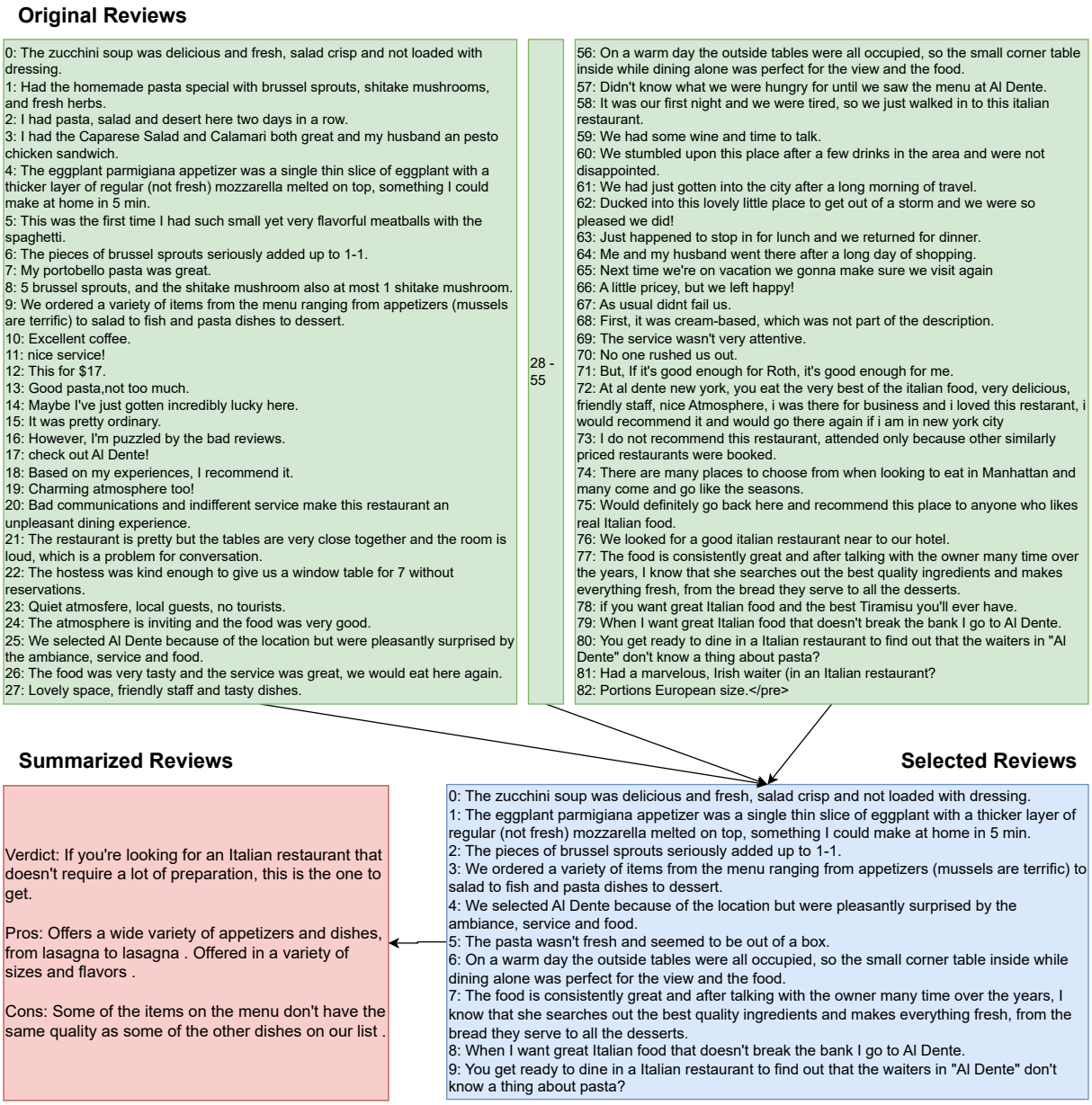

Figure 3: Example of SELSUM output.

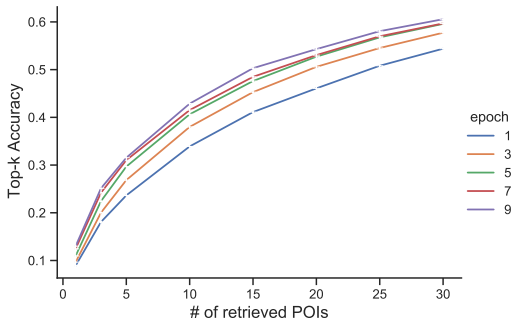

Figure 4: Top-k accuracy with varying numbers of training epochs.

Module	Acc@3	Acc@5	Acc@30	MRR
LAMB Loc	**24.83**	**32.51**	**60.92**	**0.220**
Geo-loc	22.01	29.54	58.24	0.204
Geo-dist	20.25	28.00	58.43	0.189

Table 9: Comparison between LAMB location module and other geo-coordinate-based location/distance modules on local evaluation.

weighted sum of the bi-encoder similarity score and distance score:

$$\text{sim}(p, q) = (1 - \lambda)\text{sim}(r_p, r_q) - \lambda(\text{dist}(p, q))$$

We negate the distance score to ensure the closer

the two locations, the higher the similarity. $\lambda \in [0, 1]$ is a distance score weight, where $\lambda = 0$ means the model does not consider distance at all and $\lambda = 1$ means the model computes scores by distance only.

We compare our location module with these straightforward geo-coordinate-based location/distance modules in Table 9. From the table we can clearly see that our module is much better than the alternatives.