A Proofs of Lemma 1 and Corollary 2

To derive the proof, we recall both Lemma 1 and the Ranking Probability Loss:

$$\mathcal{L}^{RPL} = -\sum_{i=1}^{|\mathbb{Q}_{tr}|}\sum_{j=1}^{N}\left(\sum_{k\in\mathbb{L}_j}[\boldsymbol{y}_i]_k\right)\log\left(\text{SoftMax}\left(\sum_{k\in\mathbb{L}_j}[\boldsymbol{f}_{\boldsymbol{q}_i}]_k\right)\right),$$

where $\tilde{s}_{ij} = \sum_{k\in\mathbb{L}_j}[\boldsymbol{f}_{\boldsymbol{q}_i}]_j$ is the modified score associated with the query-item pair $(\boldsymbol{q}_i, \boldsymbol{k}_j)$. The following Lemma sheds light on the relationship between RPL and ranking probability distribution.

Lemma. ***(Ranking Probability Loss)*** *Let $\mathbf{P} \in \mathbb{R}^{N\times N}$ denote a matrix with entries p_{jk} given by $p_{jk} = \text{Prob}\,(\textit{ranking item } \boldsymbol{k}_j \textit{ at location } k) \triangleq C\sum_{\ell\in\mathbb{L}_k}[\boldsymbol{f}_{\boldsymbol{q}_i}]_\ell$, where C is a normalizing constant. Then, the Ranking Probability Loss maximizes the probability of ranking queries $\mathbb{K}_{\boldsymbol{q}_i}$ in the ordering of the ground-truth ranking $\boldsymbol{y}_i$.*

Proof. Given $\boldsymbol{f}_{\boldsymbol{q}_i}$, the ground-truth scores $\boldsymbol{y}_i$, and a candidate item $\boldsymbol{k}_j$, we define the set $\mathbb{L}_j = \{k \in \{1, N\} : [\boldsymbol{y}_i]_k < [\boldsymbol{y}_i]_j\}$, *i.e.,* $\mathbb{L}_j$ comprises the indices k for which the ground truth score of $[\boldsymbol{y}_i]_j$ is larger than the ground truth score at location k. Additionally, without loss of generality, we assume that $\boldsymbol{f}_{\boldsymbol{q}_i}$ and $\boldsymbol{y}_i$ are normalized to have entries that lie in $[0, 1]$. We have:

$$p_{jk} \;=\; \text{Prob}\,(\text{ranking item } \boldsymbol{k}_j \text{ at location } k) \triangleq C\sum_{\ell\in\mathbb{L}_k}[\boldsymbol{f}_{\boldsymbol{q}_i}]_\ell,$$

Where C is an appropriately chosed constant, such that $\mathbf{P}$ is a matrix with entries summing to one. Further, in the context of ranking with the logits/scores, we have:

$$\begin{aligned} p_{jk} &= \text{Prob}\,(\text{ranking item } \boldsymbol{k}_j \text{ at location } k) \\ &= \text{Prob}\,(\text{logit of item } \boldsymbol{k}_j > \{\text{logits of all items } \boldsymbol{k}_k \text{ such that } k \in \mathbb{L}_k\}). \end{aligned}$$

Since the entire analysis is presented in the context of a each query $\boldsymbol{q}_i$, for convenience, we abuse notation, and denote $[\boldsymbol{f}_{\boldsymbol{q}_i}]_j = [\boldsymbol{f}]_j = f_j$. Then, we have

$$\begin{aligned} p_{jk} &= \text{Prob}\,(\text{logit of item } \boldsymbol{k}_j > \{\text{logits of all items } \boldsymbol{k}_k \text{ such that } k \in \mathbb{L}_k\}) \\ &= \text{Prob}\,(f_j > \{f_k \text{ for all } k \in \mathbb{L}_k\}) \\ &= \text{Prob}\left(f_j > \max_{k\in\mathbb{S}_k}\{f_k\}\right) \\ &\approx \text{Prob}\left(f_j > \sum_{\ell\in\mathbb{L}_k} f_\ell\right), \end{aligned} \tag{3}$$

where the last step is a consequence of the norm-bound $\|\boldsymbol{f}\|_1 \leq \|\boldsymbol{f}\|_\infty$. First, we note that, given the condition in Equation 3, for all k ranked higher than j, $p_{jk} = 0$. Further, for all k ranked lower than j, it suffices to show that the probability p_{jk} is non-zero, for $j = k$. To verify this, consider two probabilities p_{ik_1} and p_{ik_2}, with k_1 ranked higher than k_2. Let $\mathbf{1}_{ik_1}$ denote the indicator of the event associated with p_{ik_1}. We have

$$p_{ik_2} \approx \text{Prob}\left(f_j > \sum_{\ell\in\mathbb{L}_{k_2}} f_\ell \;\middle|\; \mathbf{1}_{ik_1}\right)\text{Prob}(\mathbf{1}_{ik_1}) = p_{ik_1}$$

To build intuition for the this, assume without loss of generality that $\boldsymbol{f}$ have been sorted in a non-increasing manner. Let $j = 1$ $k_1 = 1$ and $k_2 = 3$. Then, it is clear to see that

$$\begin{aligned} &\text{Prob}\left(f_1 > f_4 + f_5 + \ldots \;\middle|\; f_1 > f_2 + f_3 + f_4 + f_5 + \ldots\right)\text{Prob}\,(f_1 > f_2 + f_3 + \ldots) \\ &= \text{Prob}\,(f_1 > f_2 + f_3 + \ldots). \end{aligned}$$

Given $\mathbf{P}$ built as described above, and $\mathbf{P}^*$ defined similarly over scores $\boldsymbol{y}$, from the discussion above, we see that minimizing the distance between $\mathbf{P}$ and $\mathbf{P}^*$ is equivalent to minimizing the distance between the vectors $\boldsymbol{p} = \text{Diag}\,(\mathbf{P})$ and $\boldsymbol{p}^* = \text{Diag}\,(\mathbf{P}^*)$. We observe that, when normalized, the entries of $\boldsymbol{p}$ correspond to the probability of ranking item $\boldsymbol{k}_j$ and rank j, given sorted vectors $\tilde{\boldsymbol{f}}$ and $\tilde{\boldsymbol{y}}$. Then, the RPL is the binary cross entropy loss defined between $\boldsymbol{p}$ and $\boldsymbol{p}^*$, and represents minimizing the KL-divergence between the predicted ranking probability distribution $\tilde{\boldsymbol{f}}$ and the ground-truth ranking distribution $\tilde{\boldsymbol{y}}$, up to a normalizing constant factor. This completes the proof of Lemma 1. □

Proof of Corollary 2: To link the Ranking Probability Loss to the ListNet loss [21], we recall that:

$$\mathcal{L}^{LN} = -\sum_{i=1}^{|\mathbb{Q}_{tr}|}\sum_{j=1}^{N} P_{\boldsymbol{y},j}\log\left(P_{\boldsymbol{s},j}\right),$$

where $P_{\boldsymbol{x},j} = \dfrac{\Phi\left([\boldsymbol{x}]_j\right)}{\sum_{\ell=1}^{N}\Phi\left([\boldsymbol{x}]_\ell\right)}$ Setting $[\tilde{\boldsymbol{s}}_{\boldsymbol{q}_i}]_j = \tilde{s}_{ij} = \sum_{\ell\in\mathbb{L}_j}[\boldsymbol{f}_{\boldsymbol{q}_i}]_\ell$, and $[\tilde{\boldsymbol{y}}_i]_j = \tilde{y}_{ij} = \sum_{\ell\in\mathbb{L}_j}[\boldsymbol{y}_i]_\ell$ and a Φ that result in mapping $\tilde{\boldsymbol{s}}$ and $\tilde{\boldsymbol{y}}$ to be valid probabilities distributions, we see the equivalence between the ListNet loss, and the proposed Ranking Probability Loss.

B The CROSS-JEM Inference

The procedure for forward pass of a CROSS-JEM model during training as well as inference is outlined in Algorithm 2, while the procedure to obtain the attention map over the item union set $\mathbb{T}$ is described in Algorithm 1.

Algorithm 1 Method to create attention masks for item k_i. **Input**: T_q: Tokenized Query, T_U: Tokens in item Union Set, k: tokenized k_i keyword. **Output**: AttMask: Attention Mask for the keyword

```
procedure GetKUAttentionMask(T_q, T_U, k)
    AttMask ← []
    for i from 0 to LEN(T_q) - 1 do
        AttMask.AddItem(1)
    end for
    for i from 0 to LEN(T_U) - 1 do
        if T_U[i] in k then
            AttMask.AddItem(1)
        else
            AttMask.AddItem(0)
        end if
    end for
    return AttMask
end procedure
```

C Datasets

MS MARCO: MS MARCO document re-ranking dataset contains queries and their clicked web pages consisting of webpage title, URL, and passage. We create a śhort-textv́ersion of the dataset by considering only the titles of the webpages, making the length statistics of the dataset better aligned with the real-world applications of ranking in sponsored search. We experiment with HDCT [42] retriever based training dataset consisting of 0.37M training queries and top 10 predictions from HDCT along with their ground truth click labels. We use the dev set to report our metrics as this set has ground truth labels available for evaluation. We use $\sim$3.7M training pairs available in MS MARCO HDCT dataset sourced from [42] as described above for training CROSS-JEM and baselines using DistilBERT [43]. The target scores for all training pairs are obtained from a monoBERT model trained on binary ground truth click data with BERT-Base [44] as the base encoder for MS MARCO dataset.

SODQ: Stack Overflow Duplicate Questions dataset involves ranking questions on Stack Overflow as duplicates or not with the tags Java, JavaScript and Python [41]. It is also one of the re-ranking datasets on the popular MTEB benchmark [45]. StackOverflowDupQuestions on MTEB benchmark is the only shoft text re-ranking dataset with both training and evaluation data available, and is hence used in our experiments. Similar to our experiments on MS MARCO dataset, the target scores for training CROSS-JEM as well as baselines are obtained from a BERT [44] based monoBERT model trained on binary relevance of duplicate questions.

Sponsored Search Dataset: The training dataset for sponsored search query to advertiser matching task is created using a BERT-Large based monoBERT model trained on manually labeled and good-click data as the teacher model. A query-item (advertiser bid keyword) pair in the good click data is

Algorithm 2 Getting relevance scores for a query and retrieved set of items using CROSS-JEM. **Input**: Query q, retrieved set of N items for q where $K_q = \{k_0, k_1, ..., k_{N-1}\}$. **Output**: Scores $S = \{s_0, s_1, ..., s_{N-1}\}$

$T_q \leftarrow$ TOKENIZE(Q) ▷ Tokenize the query
$T_U \leftarrow \{\}$
$K_{tokens} \leftarrow []$ ▷ Store tokenized items
for k in K_q **do**
    $k_{tokens} \leftarrow$ TOKENIZE(k)
    $T_U \leftarrow$ UNION(T_U, k_{tokens})
    K_{tokens}.ADDITEM(k_{tokens})
end for
$T_U \leftarrow$ SORTED(T_U)
KUAttMask $\leftarrow []$ ▷ KUAttMask: item Union Attention Mask
for i from 0 to $N-1$ **do**
    AttMask $\leftarrow$ GETKUATTENTIONMASK$(T_q, T_U, K_{tokens}[i])$ (cf. Algorithm 1)
    KUAttMask.ADDITEM(AttMask)
end for
sepToken $\leftarrow$ TOKENIZE$([SEP])$ ▷ Token id for [SEP] token
encInpToks $\leftarrow T_Q$ ▷ Tokens to be passed through the Encoder
encInpToks.ADDITEM(sepToken)
for t_U in T_U **do**
    encInpToks.ADDITEM(t_U)
end for
$E \leftarrow$ ENCODER(encInpToks)
$S \leftarrow$ SELECTIVEPOOLING(E, KUAttMask) ▷ $S \in \mathbb{R}^{N \times d}$
$S \leftarrow$ CLASSIFIER(S) ▷ $S \in \mathbb{R}^{N}$
return S

obtained when the user clicked on the ad corresponding to an advertiser keyword in response to their query, and did not close the ad quickly indicating they found it relevant. This BERT-Large teacher model was used to score 100 predicted items each for 18.6M queries on the search engine during a time period. This resulted in around 1.8B query-item pairs with scores in 0 to 1 range as training data for CROSS-JEM and all baselines in Table 7.

D Metrics

- **Mean Average Precision (MAP)**: This is a ranking metric defined as the mean of Average Precision (AP) over the positive and negative detected classes:

$$\frac{1}{|Q|}\sum_{u=1}^{|U|}\left(\frac{1}{m}\sum_{k=1}^{N}P_u(k)\cdot rel_u(k)\right) \tag{4}$$

  where $|Q|$ is the total number of queries, $P_u(k)$ is the precision at cut-off k in the list, $rel_u(k)$ is an indicator function equaling 1 if the item at rank k is a relevant document, otherwise zero.

- **Mean Reciprocal Rank (MRR)**: Rank is defined as the position of the first relevant item in the ranked list. MRR is hence defined as below:

$$\frac{1}{|Q|}\sum_{i=1}^{|Q|}\left(\frac{1}{rank_i}\right) \tag{5}$$

- **Accuracy**: Positive, Negative, and Overall Accuracy denote the proportion of positive, negative, and overall instances, respectively, in the test set that are accurately identified.

- **Area Under the ROC Curve (AUC-ROC)**: The ROC curve is a plot of True Positive Rate (TPR) or sensitivity against False Positive Rate (FPR) at different thresholds.

Table 4: Improvement in performance of a large Seq2Seq model, RankT5-base after fine-tuning on short-text ranking benchmarks

Method	SODQ		MS MARCO	
	MAP@5	MAP@10	MRR@5	MRR@10
RankT5-base (pre-trained)	45.66	49.47	27.87	29.75
RankT5-base (fine-tuned)	55.9	56.8	33.72	35.14

Table 5: Hyperparameters used in CROSS-JEM.

Hyperparam	SODQ	MS MARCO	Sponsored Search
N	30	10	100
L_u	265	360	262

Table 6: (a) Ablation on loss function in CROSS-JEM; (b) Adding item token sequence information in CROSS-JEM via positional encodings in the pooling layer.

Method	BCE	CE	ListNet	RPL
MRR @10	31.46	32.03	30.27	35.45

(a)

Method	Without Positional Encodings	With Positional Encodings
MRR@10	35.45	35.68
MRR@5	33.82	33.98

(b)

E Baselines and Hyperparameters

For the monoBERT, DPR, and ANCE baselines, we tune the following hyperparameters based on the validation set accuracy: (a) learning rate; (b) weight decay, and (c) number of epochs. ColBERT is trained and evaluated with the default set of hyperparameters provided by Khattab and Zaharia [19][4]. We use the pre-trained checkpoint[5] and code-base[6] provided by Su et al. [37] for the INSTRUCTOR model, and test its zero-shot performance using the instructions mentioned in the paper. CROSS-JEM is trained with exactly same setting as monoBERT: learning rate of 1e-4, linear learning rate scheduler, and AdamW optimizer. Hyperparameters specific to CROSS-JEM (N and L_u) are provided in Table 5. The metrics for BM25 on SODQ are taken from [41], while they are computed following the procedure presented by Trotman et al. [46] on the MS MARCO dataset.

Fine-tuning RankT5-base (24L) for Short-text Ranking: We fine-tune the model checkpoint available from Zhang et al. [15] on short-text ranking benchmarks and note the performance improvements in Table 4.

E.1 Compute

All baselines on MS MARCO and SODQ datasets including CROSS-JEM were trained on 8 V100 GPUs. Experiments on proprietary Sponsored Search dataset were conducted on larger GPU cluster with 16 V100s.

F Ablation Experiments on Public Datasets

Loss Function: We demonstrate performance comparison of pointwise and listwise loss functions with CROSS-JEM architecture in Table 6 (a). While listwise loss functions (Cross-Entropy and ListNet [21]) either perform similar to or slightly better than pointwise losses (such as Binary Cross-Entropy), CROSS-JEM trained with RPL performs better than pointwise or listwise losses.

Incorporating Token Sequence Order Information: CROSS-JEM encodes the union of tokens from all ranking items, which enables fast inference by reducing the input sequence length. However, this also discards the original token order information within each item, treating them as bags of tokens. Though in our preliminary experiments, we observe that the loss of sequence information in the encoder has a negligible impact on the accuracy particularly in short-text ranking tasks (within 1%, refer Appendix G). However, token sequence information could be crucial in many ranking scenarios, and could get completely ignored in CROSS-JEM. To address this limitation, we propose an extension of CROSS-JEM that leverages sinusoidal positional encodings [47] to inject the item sequence information back into the model via the selective pooling layer. The key idea is to preserve the original position indices of the tokens for each item in the ranking list, remove them during the encoding process, and then add them back to the corresponding token context vectors after the CROSS-JEM encoder. The positional encodings are computed based on the original position indices and are summed with the token context vectors. The resulting vectors are then pooled together for classification as described in Section 3. This simple yet effective technique improves the MRR@10 by 0.2% on the MS MARCO dataset (cf. Table 6 (b)), without any additional latency. This technique could also be used to improve CROSS-JEM's performance on long texts, which we leave for future work.

G Experiments: Sponsored Search Dataset

In large-scale search and recommendation systems like sponsored search, the ranking model serves to weeding out bad retrievals and rank the prediction pool of different retrievers to select the top-k. We evaluate the effectiveness of CROSS-JEM on this real-world task of matching user queries to relevant advertiser-bid keywords. A large scale dataset consisting of 1.8B query-keyword pairs was created by mining search engine logs (detailed in Appendix C).

Accuracy comparison: As shown in Table 7, CROSS-JEM improves over the in-production sparse neural model MEB [17] in MAP by over 13%. Further, CROSS-JEM also outperforms ANCE and TwinBERT by large margins in MAP, Precision, and Recall. We also assess CROSS-JEM's ability to eliminate irrelevant items while preserving relevant ones. Table 7 presents the negative and overall accuracy when retaining top 80% of positive items per query. CROSS-JEM achieves 99.45% negative accuracy, removing nearly all irrelevant items.

Efficiency Gains: We observe that CROSS-JEM takes only 9.8 ms to score 700 keywords for a query on a A100 GPU. In contrast, monoBERT takes 41.3 ms for the same task, rendering it unsuitable for online deployment. This represents a *more than 4-fold reduction in latency* compared to standard cross-encoder models. The latency gains are because CROSS-JEM scores multiple items for a query in one shot by passing their concatenated tokens through the model. On the other hand, monoBERT scores each query-item pair independently necessitating 700 passes compared to CROSS-JEM's 7 passes. Additionally, CROSS-JEM provides 3× lower latency on GPUs than MEB on CPUs. This highlights CROSS-JEM's ability to leverage GPU acceleration for efficiency, unlike sparse models.

CROSS-JEM achieves a high throughput of 17,200 query-keyword pairs per second. This is over 5× more than the 3,350 pairs per second for monoBERT. The massive throughput and latency gains show CROSS-JEM's ability to meet the computational demands of large-scale industrial systems without sacrificing accuracy.

Understanding CROSS-JEM Efficiency Gains: To better understand **the efficiency gains** in CROSS-JEM, we analyze the effect of the significant token overlap amongst candidate items in the

[4] https://github.com/stanford-futuredata/ColBERT

[5] https://huggingface.co/hkunlp/instructor-base

[6] https://github.com/xlang-ai/instructor-embedding

Table 7: Comparison of CROSS-JEM with production baselines on Sponsored Search Ads Dataset for ranking advertiser-bid keywords for a user query. CROSS-JEM outperforms baseline methods (with a latency small enough to be deployed for real-time ranking) by 13% in MAP. CROSS-JEM filters >90% irrelevant predictions at a threshold which retains 80% of good predictions.

Method	MAP@100	P@50	R@50	AUC	Negative Accuracy	Overall Accuracy
ANCE	78.39	41.94	94.02	89.84	86.19	85.55
TwinBERT	83.56	43.50	95.36	92.10	90.60	88.58
MEB	84.38	42.94	94.65	91.77	84.59	84.40
CROSS-JEM	**97.48**	**45.76**	**99.07**	**99.41**	**99.45**	**95.27**

set $\mathbb{K}_{\boldsymbol{q}}$ for a given query $\boldsymbol{q}$. We trained an ANCE [35] dense retriever on the same train set as above. For a query $\boldsymbol{q} \in \mathbb{Q}_{te}$, recall that $\mathbb{K}_{\boldsymbol{q}} = \{\boldsymbol{k}_1, \boldsymbol{k}_2, \ldots, \boldsymbol{k}_N\}$ are the top N (=100 in our experiments) items retrieved using ANCE. Let $\mathbb{T}^L_{\boldsymbol{k}_j}$ denote word-piece tokens in $\boldsymbol{k}_j$ with a max-length of L. Let $\mathbb{T}_U$ denote the union of all tokens of items $\boldsymbol{k}_j \in \mathbb{K}_{te}$. We compute the following statistics:

$$m = \frac{1}{|\mathbb{Q}_{te}|} \sum_{\boldsymbol{q} \in \mathbb{Q}_{te}} \left(\sum_{j=1}^{N} |\mathbb{T}^L_{\boldsymbol{k}_j}| \right), \quad \text{and} \quad N_u = \frac{1}{|\mathbb{Q}_{te}|} \sum_{\boldsymbol{q} \in \mathbb{Q}_{te}} \left(\left| \bigcup_{j=1}^{N} \mathbb{T}^L_{\boldsymbol{k}_j} \right| \right),$$

where m is the *mean total tokens* and N_u is the mean size of the set $\mathbb{T}_U$. Intuitively, m is the the sum of item token lengths on average, while N_u is the cardinaltity of the union set, averaged over the queries $\boldsymbol{q}$ for which the candidate items $\mathbb{K}_{\boldsymbol{q}}$ were obtained. If the items $\boldsymbol{k}_i$ have significant overlap, $m \gg N_u$. Statistically, N_u is found to be at least $5\times$ smaller than m, indicating high redundancy, and correlates with observations on the *sponsored search data* (cf. Appendix Table 8 (b)).

Online A/B testing: CROSS-JEM was deployed in the ranking stage of a premier search engine to conduct A/B tests on live traffic. The ranking stage receives an average of 700 keywords and up to 1400 keywords in the 99th percentile, from a suite of retrieval algorithms. The control group consisted of a proprietary combination of late interaction, dense retrieval, and sparse-neural-network algorithms. CROSS-JEM demonstrated a decrease in the quick-back-rate (users who close the ad quickly, indicating non-relevance) by over 1.8%. Furthermore, as judged by expert judges, CROSS-JEM improved the proportion of accurate predictions by 10.2%.

We compare CROSS-JEM against methods that can be deployed for real-time ranking including ANCE, MEB, and TwinBERT. TwinBERT is a lighter version of ColBERT. It applies an MLP layer to individual query and keyword embeddings, unlike ColBERT which considers interactions along all token embeddings. This makes TwinBERT more efficient and practical for real-world systems due to lower storage requirements. Table 7 shows the comparison of CROSS-JEM with baseline methods in production where CROSS-JEM outperforms existing methods by large margins.

Ablation on Number of items per Query (N): From Table 9, we vary the number of items scored per query from 10 to 100. With more items, the token overlap increases, providing CROSS-JEM more opportunity for joint modeling. Correspondingly, we observe gains in negative accuracy and AUC as items per query increase.

Ablation on Encoder $\mathcal{E}_{\boldsymbol{\theta}}$ in CROSS-JEM: Table 10 shows that even with a smaller encoder, CROSS-JEM provides significant accuracy gains over sparse models like MEB while having low latency.

Ablation on Sequence Information: We analyze the effect of the ordering/sequencing of the item text on classification accuracy using a cross-encoder model; and a train dataset consists of about 100M query-item pairs $(\boldsymbol{q}, \boldsymbol{k})$, drawn from $\mathbb{Q}_{tr} \times \mathbb{I}_{tr}$, mined from proprietary search engine logs. Given $(\boldsymbol{q}, \boldsymbol{k})$, we compare two cross-encoder $\mathcal{E}$ models: 1) $\mathcal{E}_{CE}$: Standard cross-encoder scoring the pairs $(\boldsymbol{q}, \boldsymbol{k})$, and 2) $\mathcal{E}'_{CE}$: Cross-encoder trained to score pairs $(\boldsymbol{q}, \boldsymbol{k}')$, where the item $\boldsymbol{k}'$ is obtained by sorting the tokens in $\boldsymbol{k}$ alphabetically.

The hypothesis is that, if the cross-encoders $\mathcal{E}_{CE}$ and $\mathcal{E}'_{CE}$ have similar scoring accuracy, then the sequence ordering is relatively less informative for this task. While testing $\mathcal{E}'_{CE}$ is evaluated on $(\boldsymbol{q}, \boldsymbol{k}')$ pairs $\boldsymbol{k}'$ is drawn from $\mathbb{I}_{te}$, with its tokens sorted alphabetically. Table 8(a) shows how the variants perform on a test set of 10M pairs. We observe that, when the sequence information in $\boldsymbol{k}$ is

Table 8: Sponsored search dataset statistics that motivate our two key insights. (a) Cross-encoder trained with with ordered tokens ($\mathcal{E}_{CE}$) and alphabetically sorted tokens for all items ($\mathcal{E}'_{CE}$). The small performance delta (between rows) indicates that sequence ordering is not critical. (b) The mean total tokens m in a retrieved item set Q_{t_e} and the mean size N_u of tokens in the union of Q_{t_e}. The ratio of m to N_u being ~ 5 indicates strong token overlap.

Algorithm	**MAP@100**
$\mathcal{E}_{CE}$	93.03
$\mathcal{E}'_{CE}$	92.76

(a)

Max-length in word-piece (L)	$\mathbf{m}$	$\mathbf{N_u}$
12	498.38	93.70
16	499.34	94.02
32	501.52	94.60

(b)

Table 9: Variation in accuracy on varying the number of items scored per query by CROSS-JEM. We observe only minor variation in changing the number of items to be ranked at inference time. This observation is useful in real-world ranking which receive item candidates from a set of retrieval algorithms and hence the number of items to be scored can vary with the query.

N	**Negative Accuracy**	**Overall Accuracy**	**AUC**
10	99.18	94.94	99.24
20	99.37	94.96	99.34
50	99.52	94.82	99.40
80	99.56	94.71	99.51
100	99.45	95.27	99.42

Table 10: Variation in accuracy with base encoder $\mathcal{E}_{\boldsymbol{\theta}}$ in CROSS-JEM.

Encoder	**Negative Accuracy**	**Overall Accuracy**	**AUC**	**Latency CPU**
TinyBERT - 2 layer	96.52	92.76	96.67	114.3
DistilBERT - 6 layer	99.45	95.27	99.61	744.7

discarded, both the mean average precision (MAP) and accuracy are within 1% of the case when the sequence information is retained.

H Qualitative Analysis

We present additional examples comparing the ranking performance of CROSS-JEM and the baseline cross encoder model in Table 11.

Table 11: A comparison of the top-5 ranked items obtained using CROSS-JEM's listwise modeling and a pointwise ranking model [2]. Relatively more *generic* (but still relevant) items are ranked higher in baseline predictions, owing to their frequency in training data, token-level matching and other biases, which are circumvented when using listwise architectures capable of evaluating all the shortlisted items in a single forward pass, and rank the more relevant (and specific) items higher.

Query	**Top-5 Ranked Item in the Proposed Approach (CROSS-JEM)**	**Top-5 Ranked Item in a Baseline Cross Encoder**
different foods of oaxaca mexico	'the foods of oaxaca' 'oaxacan cuisine' 'exploring oaxacan food' 'authentic recipes from oaxaca, mexico' '6 things you'll love about oaxaca'	'culinary tales: the kinds of food mexicans eat' 'mexican christmas foods' 'mexican cuisine' 'culture: food and eating customes in mexico' 'popular food in mexico'
what is the bovine growth hormone	'recombinant bovine growth hormone' 'what is rbgh?' 'rbgh' 'bovine growth hormone and milk : what you need to know' 'what is rbst?'	'growth hormone' 'human growth hormone' 'alternative names for growth hormone' 'human growth hormone and insulin are friends' 'growth hormone (somatotropin)'
what is the state nickname of new mexico	'what are the nicknames of the state of new mexico?' 'state nicknames new mexico - south carolina and their explanation' 'what is new mexico's nickname?' 'new mexico state names (etymology of names)' 'the state of new mexico'	meh'the state of new mexico' meh'state of new mexico' 'new mexico' 'new mexico state university' 'state of mexico'
is the elliptical bad for your knees	'does an elliptical make bad knees worse?' 'are the elliptical machines bad for your knees?' 'is an elliptical the best machine for knees that are chronically painful?' 'why does my knee hurt on an elliptical machine?' 'elliptical machine is good for osteoarthritis of the knee!'	'what exercises can help relieve knee pain?' '4 bad exercises for bad knees' 'how to treat a knee sprain' 'how to strengthen legs with bad knees' 'yoga bad for your knees, indian doctor warns'
what do you use for oxygen facial machines	'oxygen facials and other skincare services' 'oxygenating facial treatment' 'oxygen facial : home kits, beauty benefits, side effects, process' 'benefits of oxygen facial' '4 beauty - boosting benefits of oxygen facials'	'using oxygen safely' 'the oxygen machine' 'shop cpap and oxygen' 'oxygen concentrators and generators' 'oxygen concentrators & generators'