Efficient Ensemble Transformer for Accurate Answer Sentence Ranking

Anonymous ACL submission

Abstract

Large transformer models can highly improve Answer Sentence Selection (AS2) task, but their high computational costs prevent their use in many real world applications. In this paper, we explore the following research question: ***How can we make the AS2 models more accurate without significantly increasing their model complexity?*** To address the question, we propose a Multiple Heads Student architecture (MHS), an efficient neural network designed to distill an ensemble of large transformers into a single smaller model. An MHS model consists of two components: a stack of transformer layers that is used to encode inputs, and a set of ranking heads; unlike traditional distillation technique each of them is trained by distilling a different large transformer architecture in a way that preserves the diversity of the ensemble members. The resulting model captures the knowledge of heterogeneous transformer models by using just a few extra parameters. We show the effectiveness of MHS on three English datasets for AS2; our proposed approach outperforms all single-model distillations we consider, rivaling the state-of-the-art large AS2 models that have $2.7\times$ more parameters and run $2.5\times$ slower.

1 Introduction

Answer Sentence Selection (AS2) is a core task for designing efficient retrieval-based Web QA systems: given a question and a set of answer sentence candidates (*e.g.*, retrieved by a search engine), AS2 models select the sentence that correctly answers the question with the highest probability. AS2 research originated from the TREC competitions (Wang et al., 2007), which targeted large amounts of unstructured text. AS2 models are very efficient, and can enable Web-powered question answering systems of real-world virtual assistants such as Alexa, Google Home, Siri, and others.

As most research areas in text processing and retrieval, AS2 has been dominated by the use of ever larger transformer model architectures (Vaswani et al., 2017). These models are typically pre-trained using language modeling tasks on large amounts of text (Devlin et al., 2019; Liu et al., 2019; Conneau et al., 2019), and then fine-tuned on specific downstream tasks (Wang et al., 2018, 2019; Hu et al., 2020). Garg et al. (2020) achieved an impressive accuracy by fine-tuning pre-trained Transformers to the AS2 task on the target datasets. They established the new state of the art performance for AS2 using a RoBERTa$_{\text{LARGE}}$ model.

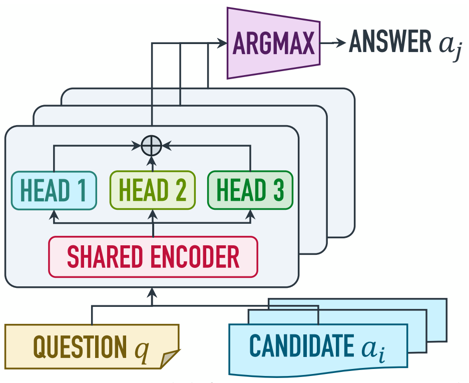

Figure 1: MHS model for answer sentence selection. The model consists of a shared encoder body and multiple ranking heads. MHS independently scores up to hundreds candidate answers a_i for question q; The one with highest likelihood is selected as answer.

Unfortunately, larger transformer models come at a cost: they require large computing resources, consume a lot of energy (critically impacting the environment (Strubell et al., 2019)), and may have unacceptable latency and/or memory usage. These downsides are critical for AS2 applications, where, for any given query, a model is required to score hundreds or thousands of candidates to select the top-k answers. Therefore, in this work, we investigate how AS2 models can be made more accurate without significantly increasing their complexity.

Previous work has addressed the general problem of high computational cost of transformer models by developing techniques for reducing their overall size while maintaining most of their performance (Polino et al., 2018; Liu et al., 2018; Li et al., 2020). In particular, Knowledge Distillation (KD) techniques have been shown to be particularly effective (Sanh et al., 2019; Turc et al., 2019; Sun et al., 2019, 2020; Yang et al., 2020; Jiao et al., 2020). KD techniques use a larger model, known as a *teacher*, to obtain a smaller and thus more efficient model, known as a *student* (Hinton et al., 2015). The student is trained to mimic the output of the teacher. However, we empirically show that, at least for AS2, BASE models trained through distillation are still significantly behind the state of the art, *i.e.*, models based on LARGE transformers.

In this paper, we introduce a new transformer model for AS2 that matches the state of the art while being dramatically more efficient. Our main idea is based on the following considerations: first, in recent years, several transformer model families have been introduced, each pretrained using different datasets and modeling techniques (Rogers et al., 2021). Second, ensembling several diverse models has shown to be an effective way to improve performance in many question answering and ranking tasks (Xu et al., 2020; Zhang et al., 2020; Liu et al., 2020; Lin and Durrett, 2020). Our contribution lies in a new approach to approximate a computationally expensive ranking ensemble into a single efficient architecture for AS2 tasks.

More specifically, our investigation proceeds as follows. First, we optimize ranking architectures for AS2 by training k student models to replicate k unique teacher architectures. When ensembled, we show that they achieve better performance than any standalone models at the cost of increased computational burden. Then, to preserve the accuracy of this ensemble while achieving lower complexity, we propose a new *Multiple Heads Student* architecture, which we refer to as MHS. As shown in Fig. 1, MHS is composed of a shared encoder body and multiple ranking heads. The encoder body is designed to derive a shared representation of input sequences, which gets fed to ranking heads. We show that if each ranking head is trained to mimic a unique teacher distribution, it is possible to achieve the desirable diversity through ensemble model while being significantly more efficient.

We train an MHS model using three different teachers: RoBERTa (Liu et al., 2019), ELECTRA (Clark et al., 2019), and ALBERT (Lan et al., 2019). We conduct experiments on three AS2 datasets: ASNQ (Garg et al., 2020), WikiQA (Yang et al., 2015), and an internal corpus (IAS2). Our results show that MHS consistently improves over all models trained with single teachers, **rivaling performance of much larger models including multiple variants of ensemble models**; further, MHS matches current state-of-the-art AS2 models (TANDA by Garg et al. (2020)), while saving 64% and 60% in model size and latency, respectively.

In summary, our contribution is four-fold:

(i) We propose MHS, an efficient architecture specifically designed to distill an ensemble of heterogeneous transformer models into a single transformer model for AS2 tasks while preserving ensemble diversity.

(ii) We conduct large-scale experiments with multiple transformer model families and show that MHS achieves better performance of equally sized distilled model, rivaling much larger ensemble and state-of-the-art AS2 models.

(iii) We discuss various training methods for MHS and show three key factors to improve AS2 performance: (*a*) multiple ranking heads in MHS, (*b*) multiple teachers, and (*c*) heterogeneity in teacher models.

(iv) We present a comprehensive analysis of the MHS, both in terms of ranking behavior and efficiency, highlighting the effect of several design decisions on its performance.

2 Related Work

2.1 Answer Sentence Selection (AS2)

Several approaches for AS2 have been proposed in recent years. Severyn and Moschitti (2015) used CNNs to learn and score question and answer representations, while others proposed alignment networks (Shen et al., 2017; Tran et al., 2018; Tay et al., 2018). Compare-and-aggregate architectures have also been extensively studied (Wang and Jiang, 2016; Bian et al., 2017; Yoon et al., 2019). Tayyar Madabushi et al. (2018) exploited fine-grained question classification to further improve answer selection. Garg et al. (2020) have achieved impressive performance by fine-tuning transformer models using a novel transfer-and-adapt technique.

2.2 Single Model Distillation

Knowledge distillation for transformer models has recently received significant attention from the NLP community. Sanh et al. (2019) presented DistilBERT, a BERT-like model with 6 layers. This student was initialized using some of the parameters from a BERT$_{\text{BASE}}$ teacher, and subsequently distilled from it. Xu et al. (2020) proposed self-ensemble/distillation methods for BERT models in text classification and NLI tasks; their teachers are obtained by ensembling student models or by averaging of model parameters from previous time steps. Turc et al. (2019) and Sun et al. (2019) also explored knowledge distillation for BERT model compression, using smaller BERT models with fewer transformer blocks as student models. Previous studies on transformer distillation have also leveraged its intermediate representation (Sun et al., 2019, 2020; Jiao et al., 2020; Mukherjee and Awadallah, 2020; Liang et al., 2020). These approaches typically lead to more accurate performance, but severely limit which pairing of teacher and students can be used (*e.g.*, same transformer family/tokenization, identical hidden dimensions).

2.3 Ensemble Distillation

Yang et al. (2020) discussed two-stage multi-teacher knowledge distillation for QA tasks. Similarly, Jiao et al. (2020) used BERT models as teachers for their proposed model, TinyBERT, in a two-stage learning strategy. Unlike their two-stage approach, our study focuses on distilling the knowledge of multiple teachers *while* preserving the individual teacher distributions. Furthermore, we explore several pretrained transformer models for knowledge distillation instead of focusing on a specific architecture. More recently, Allen-Zhu and Li (2020) formally proved that an ensemble of models of the same family can be distilled into a single model while retaining the same performance of the ensemble; however, their experiments are exclusively focus on ResNet models for image classification tasks. Kwon et al. (2020) tried to dynamically select, for each training sample, one among a set of teachers. These studies focus distillation on models that strictly share the same architecture and training strategy, which we show not achieving the same accuracy as our MHS model.

2.4 Multi-head Transformers

To the best of our knowledge, no previous work discusses multi-head transformer models for ranking problems; however, some related works exist for classification tasks. TwinBERT (Lu et al., 2020) may be the most similar approach to MHS; it consists of two multi-layer transformer encoders and a crossing layer to combine their outputs. While TwinBERT has two *bodies* which share *one classification head*, our model aims at the opposite: MHS consists of *one shared body* and *multiple ranking heads* for efficient inference and multi-teacher knowledge distillation. Another similar approach is proposed by Tran et al. (2020). However, this work exclusively focuses on non-transformer models (ResNet-20 V1 from He et al. (2016)) for image classification tasks, and is evaluated only on small datasets such as MNIST and CIFAR. Besides the different domain, this approach also focuses on distilling from architecturally similar models (distilling 50 ResNet-20 teacher models into a ResNet-20 student with 50 heads), rather than aiming at cross-model family training to increase diversity.

3 Methodology

We build up to introducing MHS by first formalizing the AS2 task (Section 3.1), and then summarizing typical transformer distillation and ensembling techniques (Section 3.2). Finally, details of the MHS approach are explained in Section 3.3.

3.1 Training Transformer Models for Answer Sentence Selection (AS2)

The AS2 task consists of selecting the correct answer from a set of candidate sentences for a given question. Like many other ranking problems, it can be formulated as a max element selection task: given a query $q \in Q$ and a set of candidates $A = \{a_1, \cdots, a_n\}$, select a_j that is an optimal element for q. We can model the task as a selector function $\pi : Q \times \mathcal{P}(A) \rightarrow A$, defined as $\pi(q, A) = a_j$, where $\mathcal{P}(A)$ is the powerset of A, $j = \texttt{argmax}_i\,(p(a_i|q))$, and $p(a_i|q)$ is the probability of a_i to be the required element for q. In this work, we evaluate MHS, as well as all our baselines, as an estimator for $p(a_i|q)$ for the AS2 task. In the remainder of this work, we formally refer to an estimator by using a uppercase calligraphy letter and a set of model parameters Θ, *e.g.*, $\mathcal{M}_\Theta$.

We fine-tune three models to be used as a teacher $\mathcal{T}_\Theta$: RoBERTa$_{\text{LARGE}}$, ELECTRA$_{\text{LARGE}}$, and ALBERT$_{\text{XXLARGE}}$. The first two share the same architecture, consisting of 24 layers and a hidden dimension of 1,024, while ALBERT$_{\text{XXLARGE}}$ is wider (4,096 hidden units) but shallower (12 layers). All three models are optimized using cross entropy loss in a point-wise setting, *i.e.*, they are trained to maximize the log likelihood of the binary relevance label for each answer separately.

While approaches that optimize the ranking over multiple samples (such as pair-wise or list-wise methods) could also be used (Bian et al., 2017), they would not change the overall findings of our study; further, point-wise methods have been shown to achieve competitive performance for transformer models (MacAvaney et al., 2019).

When training models for the IAS2 and WikiQA datasets, we follow the TANDA technique introduced by Garg et al. (2020): models are first fine-tuned on ASNQ to transfer to the QA domain, and then adapted to the target task.

Besides the three teacher models, we also train their equivalent BASE version, namely RoBERTa$_{\text{BASE}}$, ELECTRA$_{\text{BASE}}$, and ALBERT$_{\text{BASE}}$. These baselines serve as a useful comparison for measuring the effectiveness of distillation techniques.

3.2 Distilled Models and Ensembles

Knowledge distillation (KD), as defined by Hinton et al. (2015), is a training technique which a larger, more powerful *teacher* model $\mathcal{T}_\Theta$ is used to train a smaller, more efficient model, often dubbed as *student* model $\mathcal{S}_\Theta$. $\mathcal{S}_\Theta$ is typically trained to minimize the difference between its output distribution and the teacher's. If labeled data is available, it is often used in conjunction with the teacher output as it often leads to improved performance (Ba and Caruana, 2014). In these cases, we train $\mathcal{S}_\Theta$ using a *soft loss* with respect to its teacher and a *hard loss* with respect to the human-annotated labels.

To distill the three LARGE models introduced in Section 3.1, we use the loss formulation from Hinton et al. (2015), as it performs comparably to other, more recent distillation techniques (Tian et al., 2019). Given a pair of input sequence x and the target label y, it is defined as follows:

$$\mathcal{L}_{\text{KD}}(x,y) = \alpha\mathcal{L}_{\text{H}}(\mathcal{S}_\Theta(x),y) + (1-\alpha)\tau^2\mathcal{L}_{\text{S}}(\mathcal{S}_\Theta(x),\mathcal{T}_\Theta(x)) \quad (1)$$

where α and τ indicate a balancing factor and temperature for distillation, respectively. We independently tune hyperparameters $\alpha \in \{0.0, 0.1, 0.5, 0.9\}$ and $\tau \in \{1, 3, 5\}$ for each dataset on their respective dev sets. As previously mentioned, we use cross entropy as hard loss $\mathcal{L}_{\text{H}}$ for all our experiments. $\mathcal{L}_{\text{S}}$ is a soft loss function based on the Kullback-Leibler divergence $\text{KL}(p(x), q(x))$, where $p(x)$ and $q(x)$ are softened-probability distributions of teacher $\mathcal{T}_\Theta$ and student $\mathcal{S}_\Theta$ models for a given input x, that is, $p(x) = [p_1(x), \cdots, p_{|C|}(x)]$ and $q(x) = [q_1(x), \cdots, q_{|C|}(x)]$ defined as follows:

$$p_c(x) = \frac{\exp(\mathcal{T}_\Theta(x,c)/\tau)}{\sum_{j\in C}\exp(\mathcal{T}_\Theta(x,j)/\tau)} \quad (2)$$

$$q_c(x) = \frac{\exp(\mathcal{S}_\Theta(x,c)/\tau)}{\sum_{j\in C}\exp(\mathcal{S}_\Theta(x,j)/\tau)}, \quad (3)$$

where C indicates a set of class labels.

Using the technique described above, we distill three LARGE models into their corresponding BASE counterparts: *i.e.*, ALBERT$_{\text{BASE}}$ from ALBERT$_{\text{XXLARGE}}$, and so on. Furthermore, we create an ensemble of BASE models by linearly combining their outputs; hyperparameters for ensembles were tuned by Optuna (Akiba et al., 2019).

Finally, we build another ensemble model of three ELECTRA$_{\text{BASE}}$ distilled from the three LARGE models mentioned above. As we will show in Section 4, ELECTRA$_{\text{BASE}}$ outperforms all other BASE models; therefore, we are interested in measuring whether it could be used for inter transformer family model distillation. Once again, Optuna was used to tune the ensemble model.

We note that the ensemble of the three LARGE models is not used as a teacher. In our preliminary experiment, we found that the ensemble is not a good teacher, as the model was *too confident* in its prediction, a trend that is studied by Panagiotatos et al. (2019). Most softmaxed category-probabilities by the ensemble model are close to either 0 or 1 and behave like hard-target rather than soft-target, which did not improve over the KD baselines (rows 7–9) in Table 2.

3.3 Multiple-Heads Student (MHS)

As mentioned in the previous section, students trained using different teachers can be trivially ensembled using a linear combination of their outputs. However, this results in a drastic increase in model size, as well as a synchronization latency overhead, which are both undesirable properties in many applications. In this section, we introduce MHS, a transformer architecture designed to emulate the properties of an ensemble of distilled models while being more efficient. As illustrated in Fig. 2, an MHS model consists of two components: (*i*) an input encoder comprised of stacked transformer layers, and (*ii*) a set of k ranking heads, each designed to be trained with respect to a specific teacher. Each ranking head is comprised of one or more transformer layers; it receives as input the output of the shared encoder, and produces

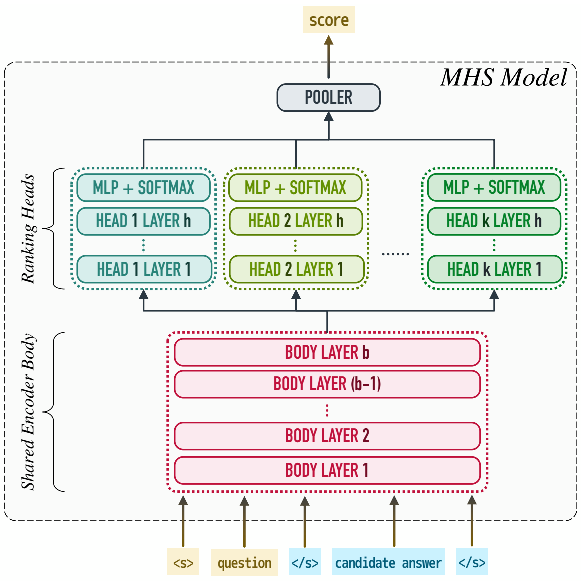

Figure 2: Detailed overview of an MHS model that consists of a shared encoder body of b transformer layers, followed by k ranking heads of h layers each; we use notation $B_b\ kH_h$ to identify an MHS configuration. **All heads are jointly trained, but each head learns from a unique teacher model**; at inference time, predictions from heads are combined by a pooler layer.

classification output. To obtain its final prediction, the MHS averages the outputs of its ranking heads. A schematic representation of MHS is shown in Figure 2.

Formally, let M_Θ be a pretrained transformer[1] of n layers. To obtain an MHS model, we first split the model into two groups: the first b blocks are used for the shared encoder body B_b, while the next $h = (n - b)$ blocks are replicated and assigned as initial states for each head H_h^i, $i = \{1, \ldots, k\}$. To compute the output for the i^{th} head, we first encode an input x using B_b, and then use it as input to H_h^i.

To train MHS, we use a linear combination of the loss function of each ranking head:

$$\mathcal{L}_{\text{MHS}}(x, y) = \textstyle\sum_{i=1}^{k} \lambda_i \cdot \mathcal{L}_i(x, y), \qquad (4)$$

where λ_i and $\mathcal{L}_i$ are the weight and loss function for the i-th head in the MHS model. Specifically, we apply the loss function of Equation 1 to each head, *i.e.*, $\mathcal{L}_i = \mathcal{L}_{\text{KD}}$ for the i^{th} head-teacher pair. We note that, while the encoder body and all ranking heads are trained jointly, each head is optimized only by its own loss. Conversely, when backpropagating $\mathcal{L}_{\text{MHS}}$, the parameters of the encoder body are affected by the output of all k ranking heads.

[1]In our experiments on ASNQ, we use a pretrained ELECTRA$_{\text{BASE}}$ model as starting point; for IAS2 and WikiQA, we use a ELECTRA$_{\text{BASE}}$ model fine-tuned on ASNQ.

Table 1: Dataset statistics. ASNQ and IAS2 contain significantly more candidates than WikiQA.

		ASNQ	IAS2	WikiQA
TRAIN	Questions	57,242	3,074	873
	QA pairs	23,662,238	189,050	8,672
	Correct answers	69,002	32,284	1,040
DEV	Questions	1,336	808	126
	QA pairs	539,210	20,135	1,130
	Correct answers	4,166	5,945	140
TEST	Questions	1,336	3,000	243
	QA pairs	535,116	74,670	2,351
	Correct answers	4,250	21,328	293

This ensures that each head learns faithfully from their teacher while the parameters of the encoder body remain suitable for the entire model.

For inference, a single score for MHS is obtained by averaging the outputs of all ranking heads:

$$\text{MHS}(x) = \frac{1}{k} \textstyle\sum_{i=1}^{k} H_h^i(B_b(x)). \qquad (5)$$

In our experiments, we use $k = 3$ heads, each trained with one of the LARGE models described in Section 3.1. We discuss a variety of combination for values of b and h; the performance for each configuration is analyzed in Section 5.3. For training, we set $\lambda_i = 1$ for all $i = \{1, \ldots, k\}$ and reuse the search space of the hyperparameters α and τ for knowledge distillation (see Section 3.2).

4 Experimental Setup

4.1 Datasets

While many studies on Transformer-based models (Devlin et al., 2019; Liu et al., 2019; Clark et al., 2019; Lan et al., 2019) are assessed for GLUE tasks (10 classification and 1 regression tasks), our interests are in ranking problems for question answering such as AS2. To fairly assess the AS2 performance of our proposed method against conventional distillation techniques, we report experimental results on a set of three diverse English AS2 datasets: WikiQA (Yang et al., 2015), a small academic dataset that has been widely used; ASNQ (Garg et al., 2020), a much larger corpus (3 orders of magnitude larger than WikiQA) that allow us to assess models' performance in data-unbalanced settings; finally, we measure performance on IAS2, an internal dataset we constructed for AS2. Compared to the other two corpora, IAS2 contains noisier data and is much closer to a real-world AS2 setting. Table 1 reports the statistics of the datasets, and more details are described in Appendix.

Table 2: Performance on IAS2, ASNQ, and WikiQA. For each metric, we highlight the **best**, 2nd **best**, and 3rd best scores. We compare our MHS model (row 14) with state-of-the-art AS2 models (Garg et al. (2020), rows 2 and 5), ensembles from distilled models (rows 10–12), and the technique proposed by Tran et al. (2020) (row 13). MHS achieves equivalent performance (Spearman ρ, $p < 0.01$) of state-of-the-art AS2 models while 2.7× smaller; it outperforms all models with a comparable number of parameters (Wilcoxon signed-rank test, $p < 0.01$).

	Teacher	Student	#params	IAS2			ASNQ			WikiQA		
				P@1	MAP	MRR	P@1	MAP	MRR	P@1	MAP	MRR
1	N/A	ALBERT$_{\text{XXLARGE}}$	222M	63.9	60.5	69.9	60.8	72.7	72.6	87.2	91.4	92.6
2	N/A	RoBERTa$_{\text{LARGE}}$ Garg et al. (2020)	335M	64.2	60.6	70.3	62.6	73.6	73.7	**89.3**	**92.6**	**93.6**
3	N/A	ELECTRA$_{\text{LARGE}}$	335M	**65.0**	**61.3**	**70.7**	**64.7**	**74.4**	**74.7**	86.7	91.0	92.2
4	N/A	ALBERT$_{\text{BASE}}$	11M	58.8	55.6	66.1	49.3	63.2	62.5	83.5	88.9	90.1
5	N/A	RoBERTa$_{\text{BASE}}$ Garg et al. (2020)	109M	59.6	56.6	67.0	54.9	67.2	67.0	82.7	88.7	89.8
6	N/A	ELECTRA$_{\text{BASE}}$	109M	62.2	58.8	68.7	61.8	71.9	72.3	86.3	90.7	91.9
7	ALBERT$_{\text{XXLARGE}}$	ALBERT$_{\text{BASE}}$	11M	61.5	57.2	68.0	56.5	68.5	68.6	84.0	89.0	90.3
8	RoBERTa$_{\text{LARGE}}$	RoBERTa$_{\text{BASE}}$	109M	63.4	59.4	69.7	62.4	72.2	72.6	83.5	89.2	90.6
9	ELECTRA$_{\text{LARGE}}$	ELECTRA$_{\text{BASE}}$	109M	63.2	**61.1**	69.6	63.7	73.9	74.1	88.1	91.6	92.9
10	Ensemble of 3 BASE (rows 4–6)		247M	63.7	59.5	69.6	62.2	72.5	72.9	88.1	91.4	92.7
11	Ensemble of 3 distilled (rows 7–9)		247M	64.2	60.0	70.1	62.7	72.8	73.1	88.1	91.7	92.9
12	Ensemble of 3 ELECTRA$_{\text{BASE}}$ distilled from $*_{\text{LARGE}}$ (rows 1–3)		327M	**65.1**	60.8	**70.8**	63.6	73.5	74.0	88.6	91.5	92.8
13	Hydra (Tran et al., 2020)		124M	63.4	59.9	69.7	62.7	73.0	73.3	88.1	91.5	92.8
14	$*_{\text{LARGE}}$ (rows 1–3)	MHS $B_{11}3H_1$ **(our approach)**	124M	64.3	60.8	70.3	**64.3**	**75.1**	**75.2**	**89.3**	**92.4**	**93.5**

4.2 Evaluation Metrics

We assess AS2 performance on ASNQ, WikiQA and IAS2 using three metrics: mean average precision (MAP), mean reciprocal rank (MRR), and precision at top-1 candidate (P@1). The first two metrics are commonly used to measure overall performance of ranking systems, while P@1 is a stricter metric that captures effectiveness of high-precision applications such as AS2.

Our models are implemented with PyTorch 1.6 (Paszke et al., 2019) using Hugging Face Transformers 3.0.2 (Wolf et al., 2020); all models are trained on a machine with 4 NVIDIA Tesla V100 GPUs, each with 16GB of memory. Latency benchmarks are executed on a single GPU to eliminate variability due to inter-accelerator communication.

5 Results

Here we present our main experimental findings. In Section 5.1, we compare MHS to state-of-the-art models and other distillation techniques using three datasets (IAS2, ASNQ, WikiQA). In sections 5.2 and 5.3, we motivate our design and hyperparameter choices for MHS by empirically validating them. Finally, in Section 5.4, we discuss inference latency of MHS comparing to other transformer models.

5.1 Answer Sentence Selection Performance

The performance of MHS on IAS2, ASNQ, and WikiQA datasets are reported in Table 2. Specifically, we compared our approach (row 14) to four groups of baselines: larger transformer-based models (rows 1–3), including the state-of-the-art AS2 models by Garg et al. (2020) (rows 2 and 5); equivalently sized models, either directly fine-tuned on target datasets (rows 4–6), or distilled using their corresponding LARGE model as teacher (rows 7–9); ensembles of BASE models (rows 10–12). We also adapted the ensembling technique of Hydra (Tran et al., 2020), which is originally designed for image recognition, to work in our AS2 setting[2] and used it as a baseline (row 13). All the comparisons are done with respect to a $B_{11}3H_1$ MHS model initialized from an ELECTRA$_{\text{BASE}}$ model: performance of other model configurations are discussed in Section 5.3. Due to the volume of experiments, we train a model with a random seed for each model given a set of hyperparameters and report the AS2 performance with the best hyperparameter set according to each dev set.

[2]Instead of 50 ResNet-20 V1 teachers paired with a 50-head ResNet-20 V1 student, we train 3 ELECTRA$_{\text{BASE}}$ teachers with different seeds and distill them into an MHS model (referred to as Hydra in Table 2) initialized from ELECTRA$_{\text{BASE}}$.

5.1.1 Vs. TANDA (BASE) & Single-Model Distillation

We find BASE models trained by TANDA (rows 4–6), the state-of-the-art training method for AS2 tasks, are further improved (rows 7–9) by introducing knowledge distillation to its 2nd fine-tuning stage. Our MHS achieves a significantly improvement over all single BASE models for all the considered datasets (Wilcoxon signed-rank test, $p < 0.01$). We empirically show in Section 5.2 that this significant improvement was achieved by both the architecture of our MHS and using heterogeneous teacher models rather than a small amount of extra parameters.

5.1.2 Vs. TANDA (LARGE)

We observe that our MHS equals (Spearman's rank correlation, $p < 0.01$) LARGE models trained by TANDA (rows 1–3), including the state-of-the-art AS2 model (Garg et al., 2020) (row 2), while the MHS has 2.7 times fewer parameters. Furthermore, our MHS consistently outperforms ALBERT$_{\text{XXLARGE}}$, which has 1.8 times more parameters than the MHS.

5.1.3 Vs. Ensembles & Hydra

For all the datasets we considered, our MHS achieves similar or better performance of much larger ensemble models, including an ensemble of ALBERT$_{\text{BASE}}$, RoBERTa$_{\text{BASE}}$, and ELECTRA$_{\text{BASE}}$ trained with and without distillation (rows 10 and 11), as well as the ensemble of three ELECTRA$_{\text{BASE}}$ models each trained using ALBERT$_{\text{XXLARGE}}$, RoBERTa$_{\text{LARGE}}$, and ELECTRA$_{\text{LARGE}}$ as teachers (row 12). We also note that MHS outperforms our adaptation of Hydra (Tran et al., 2020) (row 13), which emphasizes the importance of using heterogeneous teacher models for AS2.

5.2 Are Multiple Ranking Heads and Heterogeneous Teachers Necessary?

Using the heterogeneous teacher models shown in Table 2, we discuss how AS2 performance varies when using different combinations of teachers for knowledge distillation. The first method, KD$_{\text{Sum}}$, simply combines loss values from multiple teachers to train a single transformer model, similarly to the task-specific distillation stage with multiple teachers in Yang et al. (2020). In the second method, KD$_{\text{RR}}$, we switch teacher models for each training batch in a round-robin style; *i.e.*, the student transformer model will be trained with the first teacher model in the first batch, with the second teacher model in the second batch, and so forth.

Table 3: Comparison of single and multiple teachers distillation for ELECTRA$_{\text{BASE}}$ and MHS $B_{11}3H_1$ models on the IAS2 test set. Overall, we found that the combination of MHS architecture and use of multiple teachers is essential to achieve the best performance.

Distillation Strategy	Model	P@1	MAP	MRR
TANDA (No Teacher)	ELECTRA$_{\text{BASE}}$	62.2	58.8	68.7
Single Teacher	ELECTRA$_{\text{BASE}}$	63.2	61.1	69.6
KD$_{\text{Sum}}$	ELECTRA$_{\text{BASE}}$	63.5	60.1	69.8
KD$_{\text{RR}}$	ELECTRA$_{\text{BASE}}$	63.1	60.2	69.6
TANDA (No Teacher)	MHS $B_{11}3H_1$	62.1	59.5	68.8
Single Teacher	MHS $B_{11}3H_1$	63.2	60.5	69.5
Hydra[2] (Tran et al., 2020)	MHS $B_{11}3H_1$	63.4	59.9	69.7
One Teacher per Head	MHS $B_{11}3H_1$	**64.3**	**60.8**	**70.3**

Table 3 compares performance of the multiple-teacher knowledge distillation strategies described above to that of our proposed method; we also evaluate the effect of using one teacher per head, rather than a single teacher (ELECTRA$_{\text{LARGE}}$), on MHS. For ELECTRA$_{\text{BASE}}$, we found that KD$_{\text{Sum}}$ method slightly outperforms KD$_{\text{RR}}$; this result highlights the importance of leveraging multiple teachers for knowledge distillation in the same mini-batch. For MHS, we found that using multiple heterogeneous teachers (specifically, one per ranking head) is crucial in achieving the best performance; without it, MHS $B_{11}3H_1$ achieves the same performance of ELECTRA$_{\text{BASE}}$ despite having more parameters. Besides these two trends, the results of rows 13 and 14 in Table 2 emphasize the importance of heterogeneity in the set of teacher models.

As a result, MHS $B_{11}3H_1$ performs the best and achieves the comparable performance with some of the teacher (LARGE) models, while saving between 45% and 63% of model parameters. From the aforementioned three trends, we can confirm that the improved AS2 performance was achieved thanks to the **multiple ranking heads** in MHS, the use of **multiple teachers**, and **heterogeneity** in teacher model families; on the other hand, the slightly increased parameters compared to ELECTRA$_{\text{BASE}}$ did not contribute to performance uplift.

5.3 How Many Blocks Should Heads Have?

In Table 2, we examined the performance of an MHS model with configuration $B_{11}3H_1$; that is, a body composed of 11 blocks, and 3 ranking heads

Table 4: Performance of different MHS configurations on the IAS2 test set. Overall, we found that MHS is stable with respect to the configuration.

MHS Config	#params	P@1	MAP	MRR
$B_{11}3H_1$	124M	64.3	60.8	70.3
$B_{10}3H_2$	139M	**64.9**	**61.2**	**70.8**
$B_8\ 3H_4$	169M	64.7	60.7	70.4
$B_6\ 3H_6$	199M	64.3	60.6	70.3

with one transformer block each. In order to understand how specific hyperparameters setting for MHS influences model performance, we examine different MHS configurations in this section. Due to space constraints, we only report results on IAS2; we observed similar trends on ASNQ and IAS2. In order to keep a latency comparable to that of other BASE models, we keep the total depth of MHS constant, and vary the number of blocks in the ranking heads and shared encoder body.

Table 4 shows the results for alternative MHS configurations. Overall, we noticed that the performance is not significantly affected by the specific configuration of MHS, which yields consistent results regardless of the number of transformer layers used (1 to 6, $B_{11}3H_1$ to B_63H_6). All MHS models are trained with a combination of hard and soft losses, which makes it more likely to have different configurations converge on a set of stable but similar configurations. Despite the similar performance, we note that B_63H_6 is comprised of significantly more parameters than our leanest configuration, $B_{11}3H_1$ (199M vs 124M). Given the lack of improvement from the additional parametrization, all experiments in this work were conducted by using 11 body blocks and 1 head block ($B_{11}3H_1$).

5.4 Benchmarking Inference Latency

Besides AS2 performance, we examine the inference latency for MHS and baseline models evaluated in Section 5.1, using an NVIDIA Tesla V100 GPU. The results are summarized in Table 5. For a fair comparison between the models, we used the same batch size (128) for all benchmarks, and ignored any tokenization and CPU/GPU communication overhead while recording wall clock time. Overall, we confirm that MHS achieves a comparable latency of other BASE models. All four are within the one standard deviation of each other.

All the LARGE models including the state-of-the-art AS2 model (RoBERTa$_{\text{LARGE}}$ by Garg et al. (2020)) produce significantly higher latency, *e.g.* about 3.4 times slower than MHS; specifically, ALBERT$_{\text{XXLARGE}}$, which is comprised of 12 very wide transformer blocks, shows the worst latency among single models. Further, the latency of the two ensemble models are comparable to some of the LARGE models, thus supporting our argument that they are not suitable for high performance applications. On the other hand, our MHS model saves up to 59% latency and 62% model size compared to the ensemble model, while achieving comparable answer sentence selection performance.

Table 5: Inference Latency. For a fair comparison, batch size is set to 128 for all models. MHS achieves latency similar to those of BASE models.

Model	#params	Latency (μs)	
		avg	*std*
ALBERT$_{\text{BASE}}$	11M	2.3	0.017
RoBERTa$_{\text{BASE}}$	109M	**1.9**	0.017
ELECTRA$_{\text{BASE}}$	109M	2.0	0.018
ALBERT$_{\text{XXLARGE}}$	222M	47.0	0.370
RoBERTa$_{\text{LARGE}}$	335M	6.5	0.066
ELECTRA$_{\text{LARGE}}$	335M	6.7	0.089
Ensemble of BASE (rows 10–11)	247M	6.3	0.060
Ensemble of 3 ELECTRA$_{\text{BASE}}$	327M	6.1	0.064
MHS $B_{11}3H_1$	124M	2.6	0.020

6 Conclusions

In this work, we introduce a technique for obtaining a single efficient AS2 model from a committee of heterogeneous transformer models. This efficient approach, which we call MHS, consists of a sequence of transformer blocks, followed by multiple ranking heads; each head is trained with a unique teacher, ensuring proper distillation of the ensemble. Results show that the proposed model outperforms traditional, single teacher techniques, rivaling those of the state-of-the-art AS2 models while saving 64% and 60% in model size and latency, respectively. MHS enables LARGE-like AS2 accuracy while maintaining BASE-like efficiency.

Further analysis demonstrates that our MHS improved the AS2 performance thanks to the 3 key factors: (*i*) **multiple ranking heads** in MHS, (*ii*) **multiple teachers**, and (*iii*) **heterogeneity in teacher models**. The core idea of MHS can be extended to other tasks beside ranking, such as classification *e.g.*, GLUE (Wang et al., 2018). As such tasks come with a different objective and metrics, we leave them for future work.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631.

Zeyuan Allen-Zhu and Yuanzhi Li. 2020. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*.

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662.

Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 1987–1990, New York, NY, USA. ACM.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

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.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. TANDA: Transfer and adapt pre-trained transformer models for answer sentence selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7780–7788.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *arXiv preprint arXiv:2003.11080*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Third International Conference on Learning Representations*.

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

Kisoo Kwon, Hwidong Na, Hoshik Lee, and Nam Soo Kim. 2020. Adaptive knowledge distillation based on entropy. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7409–7413. IEEE.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E Gonzalez. 2020. Train large, then compress: Rethinking model size for efficient training and inference of transformers. *arXiv preprint arXiv:2002.11794*.

Kevin J Liang, Weituo Hao, Dinghan Shen, Yufan Zhou, Weizhu Chen, Changyou Chen, and Lawrence Carin. 2020. MixKD: Towards Efficient Distillation of Large-scale Language Models. In *International Conference on Learning Representations*.

Shih-ting Lin and Greg Durrett. 2020. Tradeoffs in sentence selection techniques for open-domain question answering. *ArXiv*, abs/2009.09120.

Dayiheng Liu, Yeyun Gong, Jie Fu, Yu Yan, J. Chen, Daxin Jiang, J. Lv, and N. Duan. 2020. Rikinet: Reading wikipedia pages for natural question answering. In *ACL*.

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 arXiv:1907.11692*.

Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. 2018. Rethinking the value of network pruning. In *International Conference on Learning Representations*.

Wenhao Lu, Jian Jiao, and Ruofei Zhang. 2020. TwinBERT: Distilling Knowledge to Twin-Structured Compressed BERT Models for Large-Scale Retrieval. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 2645–2652, New York, NY, USA. Association for Computing Machinery.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. CEDR: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1101–1104.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. XtremeDistil: Multi-stage Distillation for Massive Multilingual Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2221–2234.

Georgios Panagiotatos, Nikolaos Passalis, Alexandros Iosifidis, Moncef Gabbouj, and Anastasios Tefas. 2019. Curriculum-based teacher ensemble for robust neural network distillation. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Antonio Polino, Razvan Pascanu, and Dan Alistarh. 2018. Model compression via distillation and quantization. In *International Conference on Learning Representations*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2021. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of bert: smaller, faster, cheaper and lighter.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 373–382. ACM.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1179–1189, Copenhagen, Denmark. Association for Computational Linguistics.

Luca Soldaini and Alessandro Moschitti. 2020. The cascade transformer: an application for efficient answer sentence selection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5697–5708, Online. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. 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 4314–4323.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Multi-cast attention networks for retrieval-based question answering and response prediction. *CoRR*, abs/1806.00778.

Harish Tayyar Madabushi, Mark Lee, and John Barnden. 2018. Integrating question classification and deep learning for improved answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3283–3294, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2019. Contrastive representation distillation. In *International Conference on Learning Representations*.

Linh Tran, Bastiaan S Veeling, Kevin Roth, Jakub Swiatkowski, Joshua V Dillon, Jasper Snoek, Stephan Mandt, Tim Salimans, Sebastian Nowozin, and Rodolphe Jenatton. 2020. Hydra: Preserving ensemble diversity for model distillation. *arXiv preprint arXiv:2001.04694*.

Quan Hung Tran, Tuan Lai, Gholamreza Haffari, Ingrid Zukerman, Trung Bui, and Hung Bui. 2018. The context-dependent additive recurrent neural net. In

Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1274–1283, New Orleans, Louisiana. Association for Computational Linguistics.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the jeopardy model? a quasi-synchronous grammar for qa. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32.

Shuohang Wang and Jing Jiang. 2016. A compare-aggregate model for matching text sequences. *arXiv preprint arXiv:1611.01747*.

Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. In *ICLR 2017: International Conference on Learning Representations*, pages 1–15.

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.

Yige Xu, Xipeng Qiu, Ligao Zhou, and Xuanjing Huang. 2020. Improving BERT Fine-Tuning via Self-Ensemble and Self-Distillation. *arXiv preprint arXiv:2002.10345*.

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*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Ze Yang, Linjun Shou, Ming Gong, Wutao Lin, and Daxin Jiang. 2020. Model compression with two-stage multi-teacher knowledge distillation for web question answering system. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 690–698.

Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. *CoRR*, abs/1905.12897.

Wangshu Zhang, Junhong Liu, Zujie Wen, Yafang Wang, and Gerard de Melo. 2020. Query distillation: BERT-based distillation for ensemble ranking. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 33–43, Online. International Committee on Computational Linguistics.

A WikiQA

This dataset was introduced by Yang et al. (2015) and consists of 1,231 questions and 12,139 candidate answers. It was created using queries submitted by Bing search engine users between May 1st, 2010 and July 31st, 2011. The dataset includes queries that start with a *wh-* word and end with a question mark. Candidates consist of sentences extracted from the first paragraph from Wikipedia page retrieved for each question; they were annotated using Mechanical Turk workers. Some of the questions in WikiQA have no correct answer candidate; following (Wang and Jiang, 2017; Garg et al., 2020), we remove them from the training set, but leave them in the development and test sets.

B ASNQ

Garg et al. (2020) introduced Answer Sentence Natural Questions, a large-scale answer sentence selection dataset. It was derived from the Google Natural Questions (NQ) (Kwiatkowski et al., 2019), and contains over 57k questions and 23M answer candidates. Its large-scale (at least two orders of magnitude larger than any other AS2 dataset) and class imbalance (approximately one correct answer every 400 candidates) properties make it particularly suitable to evaluate how well our models generalize. Samples in Google NQ consist of tuples ⟨`question`, `answer`$_{\texttt{long}}$, `answer`$_{\texttt{short}}$, `label`⟩, where `answer`$_{\texttt{long}}$ contains multiple sentences, `answer`$_{\texttt{short}}$ is fragment of a sentence, and `label` indicates whether `answer`$_{\texttt{long}}$ is correct. Google NQ has long and short answers for each question. To construct ASNQ, Garg et al. (2020) labeled any sentence from `answer`$_{\texttt{long}}$ that contains `answer`$_{\texttt{short}}$ as positive; all other sentences are labeled as negative. The original release of ANSQ only contains train and development splits; We use the dev and test splits introduced by Soldaini and Moschitti (2020).

C IAS2

This is an in-house dataset, called Internal Answer Sentence Selection, we built as part of our efforts of understanding and benchmarking web-based question answering systems. To obtain questions, we first collected a non-representative sample of queries from traffic log of our commercial virtual assistant system. We then used a retrieval system containing hundreds of million of web pages to obtain up to 100 web pages for each question. From the set of retrieved documents, we extracted all candidate sentences and ranked them using AS2 models trained by TANDA Garg et al. (2020); at least top-25 candidates for each question are annotated by humans. Overall, IAS2 contains 6,939 questions and 283,855 candidate answers. We reserve 3,000 questions for evaluation, 808 for development, and use the rest for training. Compared to ASNQ and WikiQA, whose candidate answers are mostly from Wikipedia pages, IAS2 contains answers that are from a diverse set of pages, which allow us to better estimate robustness with respect to content obtained from the web.

D Do Heads Resemble Their Teachers?

To better understand the relationship between MHS's ranking heads and the teachers used to train them, we analyze the top candidates chosen by each of teacher and student models. Figure 3 shows how often each MHS head agrees with its respective teacher model. To calculate agreement, we normalize number of correct candidates heads and teachers agree on by the total number of correct answer for each head.

Intuitively, we might expect that ranking heads

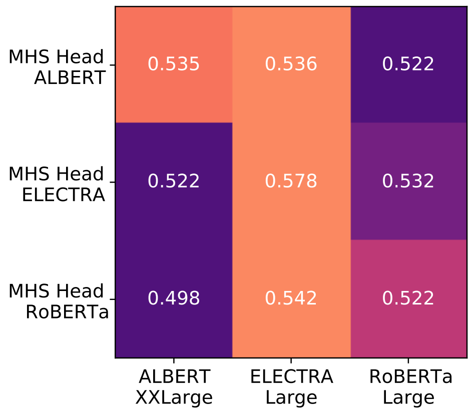

Figure 3: Agreement between each head and its teacher model for MHS. It is obtained by diving the number of correct candidates each head and teacher agree on by the total number of correct answer for each head.

would agree the most with their respective teachers; however, in practice, we notice that the highest agreement for all heads is measured with ELECTRA$_{\text{LARGE}}$. However, one should consider that the agreement measurement is confounded by the fact that all heads are more likely to agree with the head that is correct the most (ELECTRA$_{\text{LARGE}}$). Furthermore, in all our experiments, MHS is initialized from a pretrained ELECTRA$_{\text{BASE}}$, which also increase the likelihood of agreement with ELECTRA$_{\text{LARGE}}$. Nevertheless, we do note that both the head distilled from ALBERT$_{\text{XXLARGE}}$ and from RoBERTa$_{\text{BASE}}$ achieve high agreement with their teachers, suggesting that MHS ranking heads do indeed resemble their teachers.

E Common Training Configurations

Besides the method-specific hyperparameters described in Sections 3.2 and 3.3, we describe training strategies and hyperparameters commonly used to train AS2 models in this study. Unless we specified, we used Adam optimizer (Kingma and Ba, 2015) with a linear learning rate scheduler with warm up[3] to train AS2 models. The number of training iterations was 20,000, and we assess a AS2 model every 250 iterations using the dev set for validation. If the dev MAP is not improved within the last 50 validations[4], we termi-

[3]We used the warm up strategy only for the first 2.5 - 10% of training iterations, using `https://huggingface.co/docs/transformers/main_classes/optimizer_schedules#transformers.get_linear_schedule_with_warmup`

[4]For the ASNQ dataset, we considered the last 25 validations as "patience".

nate the training session. As described in Section 5.1, we independently tuned hyperparameters based on the dev set for each dataset, including an initial learning rate $\{10^{-6}, 10^{-5}\}$ and batch size $\{8, 16, 24, 32, 64\}$. Note that we train AS2 models on the ASNQ dataset for 200,000 iterations due to the size of the dataset.

For model configurations, we used the default configurations available in Hugging Face Transformers 3.0.2 (Wolf et al., 2020). For instance, the number of attention heads are 12 and 64 for ALBERT$_{\text{BASE}}$ and ALBERT$_{\text{XXLARGE}}$, 12 and 16 for RoBERTa$_{\text{BASE}}$ and RoBERTa$_{\text{LARGE}}$, and 12 and 16 for ELECTRA$_{\text{BASE}}$ and ELECTRA$_{\text{LARGE}}$, respectively. In this paper, we designed MHS leveraging the default ELECTRA$_{\text{BASE}}$ architecture, thus the number of attention heads is 12.