Unsupervised Evaluation for Question Answering with Transformers

Lukas Muttenthaler[†‡] **Isabelle Augenstein**[†] **Johannes Bjerva**[†⊙]
[†] Dept. of Computer Science, University of Copenhagen
[‡] Max-Planck-Institute for Human Cognitive and Brain Sciences, Leipzig
[⊙] Dept. of Computer Science, Aalborg University
muttenthaler@cbs.mpg.de, augenstein@di.ku.dk, jbjerva@cs.aau.dk

Abstract

It is challenging to automatically evaluate the answer of a QA model at inference time. Although many models provide confidence scores, and simple heuristics can go a long way towards indicating answer correctness, such measures are heavily dataset-dependent and are unlikely to generalise. In this work, we begin by investigating the hidden representations of questions, answers, and contexts in transformer-based QA architectures. We observe a consistent pattern in the answer representations, which we show can be used to automatically evaluate whether or not a predicted answer span is correct. Our method does not require any labelled data and outperforms strong heuristic baselines, across 2 datasets and 7 domains. We are able to predict whether or not a model's answer is correct with 91.37% accuracy on SQuAD, and 80.7% accuracy on SubjQA. We expect that this method will have broad applications, e.g., in semi-automatic development of QA datasets.

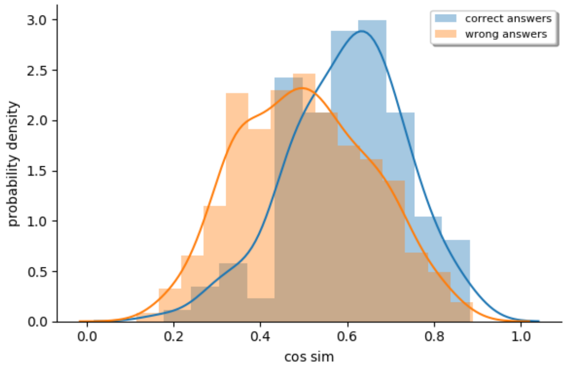

Figure 1: Probability Density Function of the cosine similarities among tokens w.r.t. the true answer span in SQuAD. Correct answer predictions (blue) tend to have higher cosine similarities than wrong answer predictions (orange).

1 Introduction

Evaluation of a QA model usually requires human-annotated answer spans to compare a model's output with. At inference time, however, it is hard to automatically estimate whether an extracted answer span is correct. While many models can provide confidence scores, and other heuristics might be used to deduce whether a prediction is correct, such measures are heavily dataset-dependent and are not likely to generalise. Hence, given a new domain, a costly procedure of human annotation needs to be initiated in order to provide an estimate of the model's accuracy. However, this approach naturally does not scale well to new unlabelled sequences.

In this work, we investigate Transformer-based QA models. We hypothesise that hidden representations of later layers in such models contain information related to correctness of answers. Indeed, we observe a consistent pattern of closely clustered answer token representations in the top three layers, whenever BERT correctly predicts an answer span (see Figure 2). Conversely, both true and predicted answer spans are clustered together with the remainder of the context, when an answer prediction is erroneous. With clustering, we refer to the transformation of high-dimensional token representations into 2-dimensional vector space through the employment of PCA (Shlens, 2014), followed by t-SNE (van der Maaten and Hinton, 2008). We furthermore see that correctly identified answer spans show a high mean cosine similarity across final layers (Figure 1). Before computing the cosine similarity between token representations, we apply PCA to remove noise, and preserve 95% of the variance in the low-rank, orthogonal representation (see 2.4 for detailed information).

We demonstrate how this insight can be used to predict whether or not a prediction from a Transformer-based QA model is correct. We evaluate our method across two distinct QA datasets in

Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 83–90
Online, November 20, 2020. ©2020 Association for Computational Linguistics

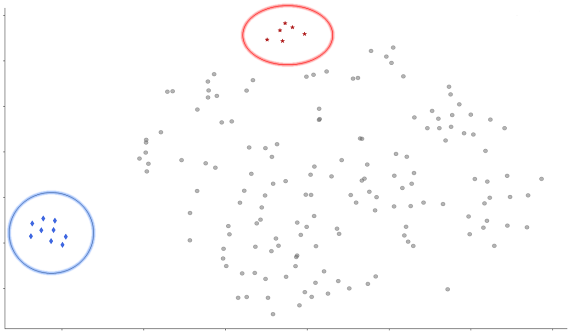

Figure 2: Hidden representations of answers are clustered separately from the remaining context for correct answer span predictions. This clustering is obtained by applying PCA followed by t-SNE (to save computational time), projecting the hidden representations for each token in a randomly chosen input sequence into $\mathbb{R}^2$. Blue diamonds: question. Red stars: answer. Grey dots: context.

English, covering 7 domains. We observe that the pattern can be used for automatic evaluation in both SQuAD v2.0 (Rajpurkar et al., 2018a) and SubjQA (Bjerva et al., 2020), a recently released dataset containing subjective questions and answers across several domains. We show that we can evaluate such models without any labelled test data, with an accuracy of 91.37% on SQuAD, and 80.7% accuracy on the more challenging and diverse SubjQA.

Contributions (i) We investigate how a Transformer-based model encodes correct and incorrect answer spans in its hidden representations; (ii) We propose a method to leverage the information contained in these representations to predict whether a given answer span prediction is correct or not; (iii) We demonstrate that a combination of our method with simple heuristics yields near-perfect predictions of answer correctness.

2 Method

2.1 Data

We experiment on two English-language QA datasets: SQuAD v2.0 (Rajpurkar et al., 2018a) and SubjQA (Bjerva et al., 2020). Since SQuAD v2.0 exclusively contains objective questions that belong to a single domain, `Wikipedia`, we contrast this with the more diverse SubjQA. SubjQA is a recently developed span-selection QA dataset that mainly consists of questions whose answer involves subjective opinions (Bjerva et al., 2020). Answer spans are extracted from review paragraphs that correspond to six different domains, namely `books`, `electronics`, `groceries`, `movies`, `restaurants`, `tripadvisor`.

2.2 Experimental Setup

For each of our implemented QA models, we use a pre-trained DistilBERT Transformer (Sanh et al., 2019) with one fully-connected output layer on top.[1] Compared to BERT (Devlin et al., 2019), with 12 layers in the base model, DistilBERT only contains 6 Transformer layers, without showing a statistically significant deterioration in performance on a variety of NLP downstream tasks (Wang et al., 2018; Rajpurkar et al., 2018a; Sanh et al., 2019).

We fine-tune BERT on either SQuAD v2.0 (Rajpurkar et al., 2018a) or SubjQA (Bjerva et al., 2020) before investigating the hidden representations. Since we analyse the similarity of hidden representations across answer span tokens, we only fine-tune BERT on answerable questions. Unanswerable questions correspond to BERT's special `[CLS]` token. Therefore, a similarity analysis of hidden representations is not carried out for these.

2.3 Answers are Separate from the Context

In order to investigate if any patterns are visible in the hidden representations, we project them into $\mathbb{R}^2$ via PCA (Shlens, 2014) and t-SNE (van der Maaten and Hinton, 2008) at each Transformer layer. This layer-wise analysis reveals how the model clusters tokens in latent space at each stage of the model. Figure 2 shows this for every token in a randomly chosen sentence pair, for which the model correctly answered the questions. Interestingly, the model clusters both the question and the answer separately from the context. We observe the same pattern with the standard BERT model, which is in line with one recent study (van Aken et al., 2019). It is this pattern which we seek to investigate further.

2.4 Answer Vector Agreements

As depicted in Figure 2, the model's hidden representations for each token in the answer span are clustered more closely in vector space for correct compared to wrong answer span predictions. This is particularly visible in the final three layers of the model, where high-level rather than low-level linguistic features are represented. To verify this observation quantitatively, we compute the average cosine similarities among all hidden representations for each token in the answer span, whenever

[1] https://huggingface.co/transformers/

the correct answer contains more than a single token. Hence, the following analysis was conducted exclusively for answerable questions since the correct answer span for unanswerable questions corresponds to the special `[CLS]` token.

Before this computation, we remove all feature representations corresponding to the special `[PAD]` token and transform the matrix of hidden representations $\mathbf{H}_i \in \mathbb{R}^{T \times D2}$ for each sentence pair sequence $(\mathbf{q}, \mathbf{c})_i$ into a lower-dimensional space to remove noise and exclusively keep those principal components that explain the most variance among the feature representations. In so doing, we use PCA (Shlens, 2014) and retain 95% of the hidden representations' variance. This yields a matrix of transformed hidden representations $\tilde{\mathbf{H}}_i \in \mathbb{R}^{T \times P}$, for each sentence pair $(\mathbf{q}, \mathbf{c})_i$. From the transformed matrix of hidden representations, we extract the matrix of hidden representations corresponding to answer span tokens $\tilde{\mathbf{H}}_{a(i)} \in \mathbb{R}^{T_a \times P}$ to compute the average cosine similarity solely across all answer vectors.

2.5 Average Cosine Similarity

The average cosine similarity among the rows of the answer representation matrix $\tilde{\mathbf{H}}_{a(i)} \in \mathbb{R}^{T_a \times P}$ is computed as follows,

$$\cos_{\tilde{H}(a)_i} = 2 \frac{\sum_j^{T_a} \sum_{k(k>j)}^{T_a} \cos(H_{a(i)}^{j^T}, H_{a(i)}^{k^T}) \in \mathbb{R}^P}{T_a T_a - T_a}, \tag{1}$$

where the cosine similarity between two non-zero vectors $\mathbf{u}$ and $\mathbf{v}$ is defined as,

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|} = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2} \sqrt{\sum_{i=1}^{n} v_i^2}} \tag{2}$$

Since the cosine similarity is a symmetric metric we can compute the sum exclusively over the upper triangular of the similarity matrix (i.e., $\forall k > j$), thus saving computational time. The computation from Equation 1 is performed for the two sets of correct and erroneous answer span predictions separately to inspect potential differences between the two w.r.t. their average cosine similarities. This was done at each Transformer layer $l \in L$, where L = 6, to examine shifts in the cosine similarity distributions across space.

2.6 Probability Distributions in Correct and Erroneous Answers

Based on having observed this pattern, we investigate how it extends to correct and incorrect answer span predictions. Figure 1 shows that the probability to observe a high internal cosine similarity, $\cos_{\tilde{H}(a)_i}$, in later layers is significantly higher for correct compared to erroneous answer span predictions.

We can formalise the pattern by investigating the cumulative distribution function of the representations (CDF). The probability $p^l_{(cdf)}$ that an observed $\cos_{\tilde{H}(a)_i}$ at Transformer layer l lies in-between two cosine values can be obtained through the following interpolation,

$$\begin{aligned} p^l_{(cdf)} &= P(\cos^l_{\tilde{\mathbf{H}}(a)} \leq \cos^l_{\tilde{H}(a)_i} + \Delta) \\ &\quad - P(\cos^l_{\tilde{\mathbf{H}}(a)} \leq \cos^l_{\tilde{H}(a)_i} - \Delta), \end{aligned} \tag{3}$$

where $\cos^l_{\tilde{\mathbf{H}}(a)}$ denotes the train distribution of $\cos^l_{\tilde{H}(a)_{(i \,\in\, N)}}$ w.r.t. either correct or erroneous answer span predictions, and Δ is a hyperparameter that refers to the boundaries of the CDF interval. Δ is set to .1 for all experiments, and $p^l_{(cdf)}$ is computed $\forall \, l \in L$.

We compare the distribution $\cos^l_{\tilde{\mathbf{H}}(a)}$ corresponding to correct and erroneous answer span predictions respectively against each other $\forall \, l \in L$. We apply independent t-tests, and adjust p-values post-hoc using Bonferroni correction to counteract the multiple comparisons problem. Analyses are performed for both development and test sets.

Table 1 shows that μ with respect to $\cos^l_{\tilde{\mathbf{H}}(a)}$ is significantly higher ($p \leq$ 1e$-$4) for correct compared to erroneous answer span predictions for Transformer layers 5 and 6 across datasets. Apart from SQuAD's test set, the same observation holds for Transformer layer 4. This is in line with both boxplots, CDFs and PDFs, and indicates that an incorrect predictions starts being erroneous at layer 4. This information can be conveniently leveraged for downstream applications, which is what we show in the following section by applying it to the evaluation of QA.

2.7 Predicting Answer Correctness

We train a simple feed-forward neural network (FFNN) with one hidden layer to predict whether the fine-tuned QA-model made an erroneous or correct answer span prediction for an input sequence x_i. The FFNN is defined as follows,

[2] T is equal to the number of tokens in a sentence pair $(\mathbf{q}, \mathbf{c})_i$ without appended `[PAD]` tokens and D = 768 which is the model's hidden size in each layer.

LAYER \ SOURCE	SQUAD				SUBJQA			
	DEV		TEST		DEV		TEST	
	p-value	diff.	p-value	diff.	p-value	diff.	p-value	diff.
LAYER 1	.121	$+.037$	.312	$+.031$	.012	$-.028$	**.000*****	$-.034$
LAYER 2	.069	$+.040$	.256	$+.021$	.020	$-.036$	.001**	$-.031$
LAYER 3	.007**	$+.054$	.419	$+.028$	.185	$-.027$	.232	$-.020$
LAYER 4	.002**	$+.061$	.422	$+.023$	**.000*****	$+.089$	**.000*****	$+.109$
LAYER 5	**.000*****	$+.151$	**.000*****	$+.094$	**.000*****	$+.115$	**.000*****	$+.133$
LAYER 6	**.000*****	$+.157$	**.000*****	$+.095$	**.000*****	$+.116$	**.000*****	$+.129$

Table 1: Differences between correct and erroneous answer span predictions with respect to $\cos_{\tilde{H}(a)}$ (see Equation 1) at every Transformer layer. p-values refer to statistically significant differences according to Bonferroni corrected independent t-tests ($p < .05$ =*, $p < .01$ = **, $p < .001$ = ***, $p = .000 \leq 1e-4$). The difference in mean cosine similarities between prediction sets is captured by the diff. column. $+$ indicates higher $\cos_{\tilde{H}(a)}$ values for correct answer span predictions. Highly statistically significant differences (***) are marked in bold face.

$$z_i = W_i^{M\times M} x_i^{M\times 1} + b_i^{M\times 1} \tag{4}$$

$$y_i = \sigma(W_i^{1\times M} z_i^{M\times 1} + b_i^{1\times 1}), \tag{5}$$

where σ denotes the sigmoid function. The sigmoid function was applied to the FFNN's raw output logits since an answer span prediction could either be correct or incorrect.

2.8 Model training

SQUAD		SUBJQA	
DEV	TEST	DEV	TEST
700	843	475	1145

Table 2: Number of examples in the leveraged development (i.e., train) and test sets.

Each FFNN is trained for a maximum number of 25 epochs until convergence. For optimization, we use Adam (Kingma and Ba, 2015) with a learning rate of $\eta = .01$ and a weight decay of .005 (this is equivalent to the L2 norm). Gradients are clipped whenever $||\frac{\partial_L}{\partial_\theta}|| \geq 10$. Input sequences are presented to the model in mini-batches of 8. The FFNN is implemented in PyTorch (Paszke et al., 2019). Both BERT for QA and the FFNN are trained and evaluated on a single Titan X GPU with 12 GB memory. Usually, a dataset is split into three parts, namely a train, a development, and a test set, where the latter two splits together comprise approx. 20-30% of the original dataset. Note that train and test datasets for SQuAD are equally large, and for SubjQA the test set even contains more than twice as many examples as the train set (see Table 2). Our train sets are the actual development sets (excluding unanswerable questions) with respect to the official QA datasets since we do not want to perform computations on hidden representations the QA model did produce during training. As such, we ascertain that the FFNN is trained on data the BERT model has never encountered during QA training. Hence, we cannot leverage development sets, and are limited to small training sets, which in turn further enhances the generalisability and potential of our approach.

We leverage one of the following three feature sets as inputs to the FFNN,

1. `Raw`. For each sentence pair, x_i, we extract $\cos_{\tilde{H}(a)_i}$ and the standard deviation (s) w.r.t. the vector of cosine similarities among the rows of the matrix $\tilde{\mathbf{H}}_{a(i)} \in \mathbb{R}^{T_a \times P}$, where $i \neq j$, at every Transformer layer. Hence, $M = 2 \times L$.
2. `Approximation`. We multiply element-wise or concatenate $p^l_{(cdf)}$ $\forall\, l \in L$ with the `raw` cosine vector (see above). However, instead of knowing to which train distribution the observed $\cos^l_{\tilde{H}(a)_i}$ belongs, we approximate $p^l_{(cdf)}$ at test time with weighting $\cos^l_{\tilde{\mathbf{H}}(a)}$ w.r.t. correct and erroneous predictions differently (see Section 2.9). Hence, $M = 2 \times L$ (weighting) or $M = 2 \times 2 \times L$ (concat).
3. `CDF-aware`. Instead of approximating $p^l_{(cdf)}$, the model is aware of whether an observed $\cos^l_{\tilde{H}(a)_i}$ must be interpolated given the distribution of correct or erroneous predictions. Again, the vector of $p^l_{(cdf)}$ $\forall\, l \in L$ is concatenated or multiplied element-wise with

the `raw` cosine vector. Hence, $M = 2 \times 2 \times L$ (concat) or $M = 2 \times L$ (weighting). This shows whether the FFNN benefits from information about the *true* train CDFs, establishing the performance ceiling when approximating $p^l_{(cdf)}$ without information loss.

2.9 Approximating CDFs

Since at test time we are not aware of whether a BERT for QA model predicted an answer span correctly or erroneously, we have implemented two different weighting strategies to approximate the true $p^l_{(cdf)} \; \forall \; l \in L$. $\cos^l_{\tilde{\mathbf{H}}(a)}$ denotes the train distribution of $\cos^l_{\tilde{H}(a)_i} \; \forall \; i \in N$ with respect to either erroneous or correct answer span predictions. Note, one must interpolate $\cos^l_{\tilde{H}(a)_i}$ given either of the two train distributions to yield $p^l_{(cdf)}$. Thus, one is required to infer to which of the two train distributions an observed $\cos^l_{\tilde{H}(a)_i}$ at test time probably belongs to. We approximated as follows,

1. `Distance`. Here, we simply compute the distance between an observed $\cos^l_{\tilde{H}(a)_i}$ to the centroid of each of the two train distributions $\cos^l_{\tilde{\mathbf{H}}(a)}$. We leveraged the inverse distance as w^l_i, such as,

$$\begin{aligned} w^l_{i(correct)} &= 1 - (\cos^l_{\tilde{H}(a)_i} - \mu^l_{i(correct)}) \\ w^l_{i(incorrect)} &= 1 - (\cos^l_{\tilde{H}(a)_i} - \mu^l_{i(incorrect)}), \end{aligned} \tag{6}$$

   where an w^l_i is higher, if $\cos^l_{\tilde{H}(a)_i}$ happens to be closer to the mean of a train distribution.

2. `CDF properties`. In this approximation, we exploited the mathematical properties of CDFs. In general, the smaller $|P(\cos^l_{\tilde{\mathbf{H}}(a)} \leq \cos^l_{\tilde{H}(a)_i}) - P(\cos^l_{\tilde{\mathbf{H}}(a)} \geq \cos^l_{\tilde{H}(a)_i})|$ is, the higher is the likelihood that an observed $\cos^l_{\tilde{H}(a)_i}$ belongs to this CDF as it denotes the area under the curve with the highest probability mass, which is considered the center. Hence, we exploited the inverse of the obtained value as w^l_i, such as,

$$\begin{aligned} &w^l_{i(correct)} = \\ &1 - |P(\cos^l_{\tilde{\mathbf{H}}(a)\;(correct)} \leq \cos^l_{H\tilde{(a)}_i}) \\ &\quad - P(\cos^l_{\tilde{\mathbf{H}}(a)\;(correct)} \geq \cos^l_{\tilde{H}(a)_i})| \\ &w^l_{i(incorrect)} = \\ &1 - |P(\cos^l_{\tilde{\mathbf{H}}(a)\;(incorrect)} \leq \cos^l_{\tilde{H}(a)_i}) \\ &\quad - P(\cos^l_{\tilde{\mathbf{H}}(a)\;(incorrect)} \geq \cos^l_{\tilde{H}(a)_i})|, \end{aligned} \tag{7}$$

   where w^l_i becomes large, the smaller $|P(\cos^l_{\tilde{\mathbf{H}}(a)} \leq \cos^l_{\tilde{H}(a)_i}) - P(\cos^l_{\tilde{\mathbf{H}}(a)} \geq \cos^l_{\tilde{H}(a)_i})|$ is.

For both approaches, we approximated the true $p_{(cdf)}{}^l_i$ through a weighted sum of $p_{(cdf)}{}^l_{i(correct)}$ and $p_{(cdf)}{}^l_{i(incorrect)}$ through the following computation,

$$\begin{aligned} &p_{(cdf)}{}^l_i = \\ &\frac{1}{2} \times (p_{(cdf)}{}^l_{i(correct)} \times w^l_{i(correct)}) \\ &+ (p_{(cdf)}{}^l_{i(correct)} \times w^l_{i(incorrect)}) \end{aligned} \tag{8}$$

In initial experiments, we examined both approximation strategies. Due to `Distance` resulting in higher macro $F1$-scores than approximating through `CDF properties`, results are reported only for `Distance`. However, the difference between the two approaches was not significant, and might require further examination in follow-up studies.

Baselines We compare the features obtained with our method against the following three baselines: (i) `Majority`, (ii) `QA concat` (hidden representations of question and answer), and (iii) `Heuristic` (e.g. n-gram overlap features).

1. `Majority`. This approach simply predicts the most common class (i.e., correct or erroneous answer span prediction).

2. `QA concat`. Concatenation of the average hidden representation w.r.t. the predicted answer span and question at last Transformer layer, where $x_i \in \mathbb{R}^{2 \times 768}$. Hence, $M = 1536$.

3. `Heuristic`. Intuitively reasonable features, where $x_i \in \mathbb{R}^9$. Hence, $M = 9$.

   (a) length of the predicted answer span;

METHOD \ SOURCE	SQUAD	SUBJQA
MAJORITY	45.65%	42.11%
QA CONCAT	63.62%	46.44%
HEURISTIC	87.23%	61.90%
$\cos_{raw}$	49.19%	69.36%
$\cos_{weight}$	63.63%	58.44%
$\cos_{concat}$	55.28%	68.14%
HEURISTIC $\oplus$ $\cos_{raw}$	87.90%	74.56%
HEURISTIC $\oplus$ $\cos_{weight}$	**88.43%**	72.29%
HEURISTIC $\oplus$ $\cos_{concat}$	88.33%	**76.42%**
$\cos_{weight}$ (CDF-aware)	78.58%	83.38%
$\cos_{concat}$ (CDF-aware)	69.14%	90.46%

Table 3: Macro $F1$-scores for the binary classification task of predicting whether a fine-tuned BERT for QA model correctly or incorrectly predicted an answer span. $F1$-scores were averaged over five different random seeds. Best scores are depicted in bold face.

(b) average n-gram overlap between predicted answer and question (i.e., BLEU score);

(c) cosine similarity between the average hidden representation w.r.t. the predicted answer span and question at last Transformer layer;

(d) vector of unigram, bigram, and trigram overlaps between predicted answer and question, normalized by the number of tokens in the answer and the question.

3 Results

3.1 Unsupervised QA Evaluation

Table 3 shows that solely exploiting $\cos_{\tilde{\mathbf{H}}(a)}$ or additionally informing the model about $p_{(cdf)}$ outperforms all baselines for two out of three approaches when evaluating on SubjQA (rightmost column). Interestingly, results slightly differ when examining QA-performance for SQuAD (centre column). The heuristics baseline yields a macro $F1$-score of 87.23%, outperforming the two other baselines by a large margin, and performing better than our proposed approaches. However, concatenating the features from the heuristics baseline with `raw` or `approximation`, further improves upon this strong baseline across both datasets, achieving 88.43% and 76.42% macro $F1$ for SQuAD and SubjQA respectively. For SubjQA, this leads to an absolute improvement of 14.5% over the strongest baseline, and suggests that information about $\cos_{\tilde{\mathbf{H}}(a)}$ is decisive to predict a QA model's answer span prediction with respect to this dataset.

The results obtained from the CDF-aware model show that further informing the model about $p_{(cdf)}$ $\forall\, l \in L$ has enormous potential to predict whether a BERT for QA model made a mistake or not. The $F1$-score of 90.46% indicates that mistakes might be predicted almost faultlessly by more sophisticated approximation methods, even without concatenating heuristic features.

Scaling $\cos_{\tilde{H}(a)_i}{}^{l}$ with $p_{(cdf)_i^l}$ appears to work better for SQuAD than concatenating $\cos^{l}_{\tilde{H}(a)_i}$ and $p^{l}_{(cdf)_i}$, whereas it is the other way around for SubjQA. Hence, combining the two information sources is crucial across datasets but which merging strategy works best is dataset dependent, and might be a function of dataset complexity or number of context domains since these are the variables in which SubjQA and SQuAD differ.

3.2 Error analysis

We investigate two examples where HEURISTIC features alone did not suffice to yield a correct prediction. Given the question from SQuAD *"What term did Eisenhower use to describe the character of communism?"*, the heuristic fails to identify the model's output as an incorrect answer. Similarly, given the question from SubjQA *"Are there any reviews on bath options at this hotel?"* and the correct model answer *"great bathroom"*, the heuristic fails to identify this as a correct answer. The concatenation of $\cos_{\tilde{H}(a)}$, and additional information about $p_{(cdf)}$ were necessary to obtain correct predictions. We further see that the observed $\cos_{\tilde{H}(a)}$ values are in line with our qualitative and statistical analyses. $\cos_{\tilde{H}(a)}$ is significantly higher in the final three Transformer layers for a correct prediction, and remains unchanged for erroneous predictions (see Table 4 for more details).

4 Related Work

Since automatic evaluation is considered an important topic in other areas of NLP, e.g. MT (Papineni et al., 2002) and summarisation (Owczarzak et al., 2012), we want to draw attention to such techniques for QA. To the best of our knowledge, ours is the first proposal unsupervised QA evaluation method.

One recent study which we take inspiration from present a layer-wise analysis of BERT's Transformer layers to investigate how BERT answers questions (van Aken et al., 2019). For each Transformer layer, they project the model's hidden representations into $\mathbb{R}^2$ to illustrate how BERT clusters

Source	SQuAD	SubjQA
Question	”What term did Eisenhower use to describe the character of communism?”	”Are there any reviews on bath options at this hotel?”
Answer	”atheistic”	”great bathroom”
BERT QA	incorrect	correct
HEURISTIC	correct ✗	incorrect ✗
HEURISTIC $\oplus \cos_w$	incorrect ✓	-
HEURISTIC $\oplus \cos_c$	-	correct ✓
$\cos^l_{\tilde{H}(a)}\ \forall l \in L$	$[0.07, 0.16, 0.26, 0.27, 0.31, 0.20]$	$[0.06, 0.14, 0.29, 0.62, 0.63, 0.55]$

Table 4: Error analysis. $\cos^l_{\tilde{H}(a)}$ is presented for each of the six Transformer layers.

different parts of an input sequence while searching for an answer span. We replicate their findings for SQuAD, and show that this insight holds across two datasets and seven domains through observing the same patterns w.r.t. SubjQA. However, we use this qualitative analysis just as an initial step from which we start extracting information to develop an unsupervised QA evaluation method.

Arkhangelskaia and Dutta (2019) investigate which tokens in sentence pairs receive particular attention by BERT's self-attention mechanisms to answer a question, and how the multi-headed attention weights change across the different layers. Similarly to van Aken et al. (2019), the authors did solely conduct a qualitative analysis of the model. Contrary to van Aken et al. (2019), the study focuses on a single implementation of BERT and exclusively exploited SQuAD (Rajpurkar et al., 2016, 2018b) without inspecting BERT's behaviour with respect to other, more challenging QA datasets where contexts belong to different domains. The latter is particularly important for real-world settings, which is why we also evaluate on SubjQA.

5 Discussion

The heuristic method we investigate in this work, based on features such as n-gram overlap between question and answer, yielded surprisingly high results on SQuAD. On the other hand, the results for SubjQA were quite low when using the heuristic only. This shows that, although a simple heuristic might be sufficient for a single dataset, it does not necessarily generalise across datasets and domains.

Conversely, our proposed method, which takes answer span similarities into account, was highly successful on SubjQA without the need for any heuristic features, but only outperformed the SQuAD baseline by 15-20% macro F1 score. Combining the two methods yielded the best results across both data sets and all domains. This demonstrates that the information contained in the heuristic approach and in our proposed method are complementary.

5.1 Error analysis

The concatenation of $\cos_{\tilde{H}(a)}$, and further information about $p_{(cdf)}$ were necessary to obtain correct predictions. The observed $\cos_{\tilde{H}(a)}$ values are in line with our qualitative and statistical analyses. $\cos_{\tilde{H}(a)}$ is significantly higher in the final three Transformer layers for a correct prediction, and remains unchanged for erroneous predictions (see last row in Table 4). It is interesting to note that for a correct answer span prediction $\cos_{\tilde{H}(a)}$ increases considerably from layer 3 to layer 4, but no notable change can be observed thereafter.

6 Conclusion

We have shown that the hidden representations of answers in transformer-based models can be used to predict whether or not that answer is correct. In combination with heuristic methods, we are able to predict the correctness of answers with a macro F1 score of 88.38% for SQuAD and 76.42% for SubjQA. Apart from the applications in unsupervised evaluation of QA, we expect that this method can be applied to semi-automatic generation of QA datasets.

Acknowledgements

The authors would like to thank the anonymous reviewers for their feedback which contributed to improving the final version of the paper.

References

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How does bert answer questions? a layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1823–1832, New York, NY, USA. Association for Computing Machinery.

Ekaterina Arkhangelskaia and Sourav Dutta. 2019. Whatcha lookin' at? deeplifting bert's attention in question answering. *CoRR*, abs/1910.06431.

Johannes Bjerva, Nikita Bhutani, Behzad Golshan, Wang-Chiew Tan, and Isabelle Augenstein. 2020. SubjQA: A dataset for Subjectivity and Review Comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Karolina Owczarzak, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization@NACCL-HLT 2012, Montrèal, Canada, June 2012, 2012*, pages 1–9. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

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.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018a. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018b. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789. Association for Computational Linguistics.

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, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The 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. *CoRR*, abs/1910.01108.

Jonathon Shlens. 2014. A tutorial on principal component analysis. *CoRR*, abs/1404.1100.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pages 353–355. Association for Computational Linguistics.