VMware's Automated Question Answering System

Rick Battle
rbattle@vmware.com
VMware R&D AI Lab (RAIL)

Omar Khattab
okhattab@stanford.edu
Stanford DAWN Lab

ABSTRACT

Keyword-based search has been outpaced by recent advances using deep Language Models (LMs) for Information Retrieval (IR) [11]. Yet VMware's internal and customer-facing portals continue to rely on traditional, keyword-based search, which fails to address many basic queries unless a user knows the exact phrase to search for. Moreover, thanks in large part to Google's Featured Snippets [13], there are growing expectations from users to get an answer for a given question directly in the search results instead of having to read through potentially hundreds of pages of documentation to find the answer. For VMware, this expectation becomes a significant burden for support engineers who frequently have to dig through dense documentation and repeatedly answer the same questions.

Using modern Language Models to power neural Information Retrieval and extractive Question Answering (QA), we can ease the burden of front-line support staff by directly answering customers' questions. In this paper, we introduce vAQA: VMware's Automated Question Answering system, which uses ColBERT for IR and ELECTRA for QA to answer questions directly from our published content. We'll show a 4.0–5.5× improvement over VMware's current, in-production search appliances.

KEYWORDS

machine learning, language modeling, information retrieval, question answering

1 INTRODUCTION

Since the introduction of the Transformer architecture [34], and the BERT [12] model in particular, the Natural Language Processing (NLP) community has been in a renaissance. When applied to the IR domain, these powerful LMs demonstrated dramatic performance gains over traditional search techniques.

IR is a critical piece of the customer experience puzzle. Traditional, keyword-based search only performs well if the customer already knows which terms to use to find the content they're looking for. Customers are far more likely to have a question, unaware of the exact search terms to use to surface the content containing the answer. A motivating hypothesis of this work is that IR paired with a QA model can provide significant benefits to the customer support community.

To wit, we are proud to present vAQA, **V**Mware's **A**utomated **Q**uestion **A**nswering system. vAQA is a pipeline of NLP technologies that come together to answer customers' questions. vAQA starts with vNLP, VMware's Natural Language Preprocessor [3], to cleanse the question. Col(v)BERT, our custom-trained version of ColBERT, then searches for passages that contain the answer to the question. Finally, ELECTRA [10] highlights the answer in the retrieved passages.

2 RELATED WORK

As of 2015, 83% of text-based IR systems used TF-IDF [7]. There are two primary search appliances in production at VMware today: Elasticsearch (ES) [9] runs search for VMware Docs and Coveo [16] runs search for My VMware. ES and Coveo utilize modernized versions of TF-IDF, but are still "just" traditional, keyword-based search engines.

This project was first conceived in 2018 when BERT showed a massive gain on the SQuAD 2.0 [27] benchmark, a challenging extractive QA task. Since this predated more recent gains in neural IR, Elasticsearch was used for IR, with BERT attempting to extract answers from the passages returned by ES. Unfortunately, the answers extracted from most of the passages were nonsensical. A deep-dive analysis into the numerous failure cases revealed two primary issues: (1) ES often returned passages that didn't contain the answer to the question, and (2) when ES did return a useful passage but comprehending the question and answer required knowledge of VMware-specific terminology (like a product name or technical jargon), BERT failed to identify the answer.

To address the second shortcoming, we introduced two innovations: (1) a preprocessing engine, vNLP, VMware's Natural Language Preprocessor, which transforms our highly technical documentation into a format suitable for ingestion into a LM, and (2) vBERT, a domain-adapted, VMware-specific version of BERT [1, 2].

Following the introduction of BERT, tremendous progress was made developing neural IR systems built on top of modern language models. This project briefly came back to life when Google released REALM [15]. It introduced a method for unsupervised training of a LM for IR. Google open-sourced the REALM codebase and we attempted to use it, but their code was specifically designed to reproduce their results with their data set. It wasn't sufficiently documented to use with a different data set. Specifically, it lacked instructions for transforming one's input text into their format for training and inference. So, the project was set aside until a more usable model/codebase came along.

Following the introduction of REALM, other modern-LM-based IR algorithms were proposed in the literature. Stanford's DAWN Lab introduced ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT [22]. Facebook released DPR: Dense Passage Retrieval for Open-Domain Question Answering [18]. Sentence Transformers adapted S-BERT [30] and MPNet [32] for semantic search [28]. We had initially set out to test all of these, but further literature review regarding DPR revealed it to be uncompetitive both in-domain and out of domain [21, 33]. There is a modified version of DPR [17] that performs well, but it relies on T5-Large [26], which we determined to be computationally intractable (it has 770M parameters compared to BERT-Base's 110M), so we abandoned DPR and proceeded with testing ColBERT and Sentence Transformers' Semantic Search.

3 VAQA

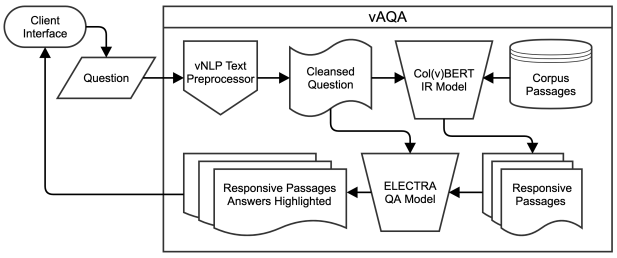

Figure 1: vAQA System Diagram

As shown in Figure 1 vAQA is a pipeline of models working together. A client interface—be it a traditional search interface, in-product chatbot, slackbot, among others—accepts a question from the customer and sends it to vAQA. The question is first cleansed with vNLP. The cleansed question is given to the IR model, Col(v)BERT, which retrieves responsive passages. The cleansed question and the retrieved passages are given to the QA model, ELECTRA, which highlights the answer to the question in the retrieved passages. The passages with answer highlights are passed back to the client for display to the customer.

3.1 Input Cleaning

Input cleansing is critical [23] for both the IR step and QA step. The difference between "what is vsphere" and "what is vshere" is the difference between getting an answer to the question and getting noise [5].

vAQA makes use of vNLP for input cleansing. Along with correcting spelling, vNLP also corrects encoding problems, decodes HTML character codes, filters noise (by masking out things like timestamps, IP/MAC addresses, hex values, etc.), splits compound words (e.g. "acceptalleulas" becomes "accept all eulas"), and expands abbreviations (e.g. "cmd" becomes "command").

vNLP was originally designed for classification, which can benefit from aggressive noise reduction. But, content that is noise in classification can be signal in IR. For example, a product's version number can generally be safely masked out for a classification task, but is very important for a query such as "how to upgrade to vSphere 6.5". So, we conducted minor tuning of vNLP's default options to mask out only true noise and keep the signal.

We didn't perform any specific experiments to gauge the importance of vNLP to IR performance, but the winning entry [35] in the vBetter[1] Zero-Shot Information Retrieval competition [4] saw a *15.3 point* boost to their score after using vNLP.

3.2 Information Retrieval

To correctly answer a question using an extractive QA model, the system must first retrieve the passage from a corpus that contains the answer. The original version of this system used Elasticsearch for the IR step. Consequently, BERT was generally unable to extract a useful answer to the question because ES was often unable to locate a passage that contained the answer. It is only because of the recent advancements in IR via the use of modern LMs that this system is functional today.

[1]vBetter is VMware's internal Machine Learning community platform where we host Kaggle-style competitions on VMware-specific problem sets.

3.3 Zero-Shot Model Training

Since we lack a VMware-specific IR training data set, the retrieval system must first be trained on a publicly available IR data set, such as MS MARCO [25], Google's Natural Questions (NQ) [24], GooAQ [19], etc.

To train ColBERT, one needs a sufficiently large set of training triples. A training triple takes the form $\langle q, p^+, p^- \rangle$, with query q, positive passage p^+, and negative passage p^-. MS MARCO provides training triples. To use either NQ or GooAQ, one must generate the negative passages. The resulting strength of the IR model strongly depends on both the diversity of types of questions and answers, along with the difficulty of the negatives. If it's too easy for the model to distinguish between the positive and negative passages, then the model learns nothing. If the selected negative passage is "too good" of a responsive passage for the query, then the model is unable to learn what a negative passage is.

Starting from the larger MS MARCO document data set, researchers at Stanford generated new training triples, creating a derivative data set named sMarco (Stanford MARCO), with longer passages for both the positive and negative passages and stronger negatives than the original MS MARCO triples.

A model trained on any of the aforementioned data sets is considered a *zero-shot* model since the training set is unrelated to our corpus and list of queries. The performance of training on those data sets (discussed in sections 4.3.2 & 4.3.3) can be improved with in-domain model supervision.

3.4 Model Supervision

Zero-shot models can be powerful, but just like when a new employee joins a company and their performance improves after learning the technical jargon of the company, an IR model's performance can be improved by training on the content they're indexing. This is achieved with in-domain model supervision.

(1) Index our content with the zero-shot model
(2) Retrieve the top 1k passages for each query
(3) Generate 40 training triples per query
  (a) For all 40 triples, the query is the query
  (b) The first 20 positive passages per query are the top-ranked passage
  (c) The second 20 positive passages per query are the second-ranked passage
  (d) All negative passages are randomly sampled from the 20th passage to the thousandth passage
(4) Train using the generated training triples
  Note: This is *not* continued training of the zero-shot model, but a fresh training cycle.

Figure 2: Procedure for performing weak supervision.

Landing Page	URL Clicks	Impressions
https://www.vmware.com/support/ws5/doc/new_guest_tools_ws.html	538	1066
https://docs.vmware.com/en/VMware-Tools/10.2.0/com.vmware.vsphere.vmwaretools.doc/GUID-391BE4BF-89A9-4DC3-85E7-3D45F5124BC7.html	187	735
https://docs.vmware.com/en/VMware-vSphere/6.7/com.vmware.vsphere.html.hostclient.doc/GUID-ED3ECA21-5763-4919-8947-A819A17980FB.html	34	362
https://docs.vmware.com/en/VMware-vSphere/5.5/com.vmware.vmtools.install.doc/GUID-391BE4BF-89A9-4DC3-85E7-3D45F5124BC7.html	34	177
https://www.vmware.com/support/ws5/doc/ws_newguest_tools_windows.html	5	71
https://www.vmware.com/support/ws45/doc/new_guest_tools_ws.html	4	7
https://www.vmware.com/support/ws5/doc/ws_newguest_tools_linux.html	3	14
https://kb.vmware.com/s/article/1002234	2	2
https://kb.vmware.com/s/article/1035392	2	2
https://kb.vmware.com/s/article/2129825	1	1

Table 1: Sample Google Search Console click data for the query, "how to install vmware tools". Note the strong exponential decay in the click count as you look down the list of results.

3.4.1 Weak Supervision. Model supervision is accomplished by using a zero-shot model to generate training triples from our content. Weak supervision, as described in [21], is a process for improving the performance of an IR model by training it using triples it itself generates. It can be thought of as a confidence boost to the language model. With a zero-shot model, the IR model is trained on a public data set. It then indexes our content, which is generally quite different from general web content. Thus, weak supervision "boosts the confidence" of the language model by reinforcing its choice of retrievals for the top and bottom search results.

This is referred to as "weak supervision", because it's only guided by the ranking ability of the zero-shot model. See Figure 2 for the steps.

(1) Index our content with a zero-shot model
(2) Retrieve the top 1k passages for each query
(3) Generate $\sum_{n=0}^{5} 2^n$ training triples per query
  (a) For each triple, the query is the query
  (b) For the positive passages:
    (i) If there is an nth page in the Google Search Console data, select the top passage from that page
    (ii) If not, select the next top passage as ranked by the zero-shot IR model
  (c) All negative passages are randomly sampled from the 20th passage to the thousandth passage
(4) Train using the generated training triples

Figure 3: Procedure for performing Google-vision.

3.4.2 Google-vision. Improving upon the concept of weak supervision, we introduce a new supervision technique, which we refer to as "Google-vision". That's because the model supervision is guided by Google Search Console click data. While inspecting the Search Console data, we noted that not all queries in the Search Console data were suitable for use, so we set a threshold based on the number of clicks and click-through rate for inclusion in the larger training set and a higher threshold for inclusion in the test set to ensure only high-quality questions were being tested. The resulting sets, dubbed vQueries, had 32k queries in the training set and 1.2k queries in the test set.

Instead of using the top and second-ranked passages as the positive examples, the positive passages are pulled from the clicked pages. Also, we observed a negative exponential relationship between pages and clicks when looking down the list of clicked pages per query. See Table 1 for sample click data.

So, instead of 20 negatives each for the top two positive passages, we used exponential decay for the number of negatives per positive, i.e. 32 negatives for the first positive, 16 for the second, 8 for the third, then 4, 2, and 1. Most queries did not have 6+ pages in the Google Search Console data, so when there were fewer, after exceeding the number of pages in the click data, positive passages were taken in the order of their ranking from the zero-shot model. Negative passage selection remains the same. See Figure 3 for the procedure for Google-vision.

The other major advantage of model supervision is, it allows different models to be used for the zero-shot supervision-triple-generation step and final-supervised-model training/ inference. vBERT-Large, having 340M parameters, is computationally intractable for inference at scale, which is why we focused on vBERT-Base for most of the project. However, we discovered that we can train vBERT-Large as a zero-shot model with sMarco, index our content, and generate the Google-vision training triples. Those training triples can then be used to supervise vBERT-Base. Since the power of a supervised model depends on the *quality* of the triples generated by the zero-shot model, not the *speed* of the zero-shot model, using a larger LM for training-triple generation and supervising a smaller model yields better results than training a smaller zero-shot model to generate triples for supervising the smaller model. See Section 4.3.4 for a discussion of the supervised-model scores.

Passage Length	Passage Overlap	Metadata	Recall@5
180	0	No	36.84
450	0	No	39.65
450	150	Yes	45.43

Table 2: Retrieval results on the vQueries test set using Col(v)BERT-Base trained on MS MARCO, experimenting with increasing the input passage length and adding metadata to each passage.

3.5 Input Corpus Optimization

A major challenge with using a modern LM for IR is its Maximum Sequence Length (MSL) limit: an input passage exceeding 512 tokens in length is truncated and any information beyond the 512th token is lost. Most pages in our corpus significantly exceed the MSL limit and therefore need to be split into passages.

Our first attempt at indexing our content used a relatively short passage length (180 tokens) because that's the length used for training with MS MARCO passages, and unexpected things can happen if a different passage length is used for inference than was used for training. We then experimented with increasing the passage length to 450 tokens, which yielded a moderate performance improvement of 2.8 points. It was at this point, that a deep-dive into the failure cases revealed what we call the "context carryover" problem.

Consider the query "how to check vcenter version". The correct page could be a KB article where the title says something about checking vCenter's version number, but the procedure that is the actual answer could be below several paragraphs of preamble, thus separating the answer itself from the context (found in the page title) that is required for the model to know that this procedure *is* the answer. Thus, some mechanism was needed to "carry the context" over to all passages that make up a single page.

Our solution to this was to introduce passage overlap and append the page's metadata to each passage. Since each passage is a maximum of 450 tokens long, there's room for metadata like page title, product name(s), version number(s), etc. The passage overlap and inclusion of metadata resulted in a more significant 5.8 point gain. See Table 2 for a breakdown of corpus enhancements vs score.

3.6 Extractive Question Answering

ColBERT doesn't have the ability to highlight the relevant words in a responsive passage (e.g. the answer to a question), so we chose to make use of an extractive QA model to highlight the answer to the question in the responsive passages.

At the time of selection, ELECTRA was the top performing single (non-ensemble) model on the SQuAD2 Leaderboard [14], 1.9 points better than human performance, and only 1.8 points behind large ensemble methods.

Having been trained on SQuAD2, the model performs exceedingly well at short answer/factoid style questions, i.e. "what is ...". It can answer longer form questions, such as "how to" type questions where the answer is a long procedure, but is more likely to struggle under those conditions. And, if the search query is not a question, e.g. "vSphere Release Notes", then nothing will be highlighted. Solving these issues is left as future work.

4 EXPERIMENTAL EVALUATION

There are plenty of publicly available IR reference data sets for testing retrieval models, but strong performance on a public data set is generally only a rough predictor of performance on a VMware-specific data set. We don't have a QA pair data set to directly test the whole system with, but what we do have is the Google Search Console data for our public-facing content. That data contains all search queries that land people on our content, the associated click-through data, and, crucially for this project, the subset of those queries that resemble questions. Since we don't have labeled answers for each question, we're limited to testing the retrieval model's ability to surface the page that contains the correct answer. Thus, the performance metric we used was Recall at various points, specifically 5, 10, 50, and 200. Recall@5 indicates the page that contains the answer was returned in the top five search results.

4.1 Queries

The queries for our test set are a subset of the questions from the Google Search Console data. After reviewing the questions for quality, we set a threshold of at least 3 clicks with a click-through rate greater than or equal to 0.5 for inclusion in the test set. 1,246 questions met those requirements. The following are a few sample queries:

(1) how to install vmware tools ubuntu
(2) how to check vcenter version
(3) what is vsphere ha

4.2 Retrieval Corpus

To ensure a fair comparison between our in-production search appliances and our system, the retrieval corpus was limited to the content indexed on VMware Docs (Elasticsearch). The corpus is comprised of Documentation, Solutions Guides, Release Notes, and KB Articles, totalling 287K pages. Retrieval on My VMware (Coveo) was filtered to the same set.

4.3 Performance Comparison

To evaluate each system's performance, we ran the test queries discussed in Section 4.1 against the corpus discussed in Section 4.2 to measure where the correct page is found in the returned search results. "Correct" in this context means the most clicked-through page from the Google Search Console data.

We did not hand review the top page(s) for each query, since that would require sifting through all 287k pages in the corpus to determine if there was a better match for any given query. Since (a) Google is widely regarded as the gold standard for web search, (b) the click-through data provides additional indication that the top page is the best page, and (c) improvements made to our system *without* influence from the click data (such as those discussed in Section 3.5) caused an increase in Recall, we found this metric and test set to be a reliable measure of retrieval quality. See Table 3 for a full breakdown of each model's performance.

4.3.1 In-Production Search Appliance Performance. Elasticsearch turned in a score of 11.56 for Recall@5, which means it was only

Method	Language Model	Training Corpus	Dedupe Version	Recall @5	Recall @10	Recall @50	Recall @200
Elasticsearch	-	-	-	11.56	13.96	21.51	27.61
Coveo	-	-	-	15.97	21.51	-	-
ColBERT	BERT-Base	MS MARCO	1	40.05	50.96	70.39	84.19
ColBERT	BERT-Base	sMarco	1	45.51	54.57	73.27	86.67
ColBERT	BERT-Base	Natural Questions	1	20.63	27.21	43.34	61.00
Col(v)BERT$_{\text{msmarco}}$	vBERT-Base	MS MARCO	1	45.43	55.62	75.68	85.79
Col(v)BERT$_{\text{msmarco-googlevision}}$	vBERT-Base	vQueries$_{\text{msmarco}}$	1	46.87	56.26	76.32	87.80
Col(v)BERT$_{\text{smarco}}$	vBERT-Base	sMarco	1	48.39	58.19	79.37	90.37
Col(v)BERT$_{\text{smarco-googlevision}}$	vBERT-Base	vQueries$_{\text{smarco}}$	1	49.04	58.83	79.21	88.68
Col(v)BERT$_{\text{naturalquestions}}$	vBERT-Base	Natural Questions	1	31.70	41.81	65.57	84.27
Col(v)BERT$_{\text{naturalquestions-googlevision}}$	vBERT-Base	vQueries$_{\text{naturalquestions}}$	1	34.75	45.10	67.50	84.83
Col(v)BERT$_{\text{gooaqir}}$	vBERT-Base	GooAQ-IR	1	37.00	49.28	70.63	84.75
Col(v)BERT$_{\text{smarco-vbertlarge}}$	vBERT-Large	sMarco	1	53.21	61.64	81.70	90.53
Col(v)BERT$_{\text{smarco-vbertlarge-googlevision}}$	vBERT-Base	vQueries$_{\text{smarco-vbertlarge}}$	1	53.37	62.84	82.02	90.37
Col(v)BERT$_{\text{smarco-vbertlarge}}$	vBERT-Large	sMarco	2	63.48	74.72	**90.21**	**94.54**
Col(v)BERT$_{\text{smarco-vbertlarge-googlevision}}$	vBERT-Base	vQueries$_{\text{smarco-vbertlarge}}$	2	**63.88**	**75.68**	88.04	93.18
msmarco-bert-base-dot-v5	BERT-Base	MS MARCO	1	34.27	44.30	67.82	82.66
multi-qa-mpnet-base-cos-v1	MPNet-Base	Many [28]	1	46.95	58.75	80.66	91.49

Table 3: Retrieval results on the vQueries test set. The subscript on vQueries training set indicates the model that was used to generate positive and negatives. For example, vQueries$_{\text{msmarco}}$ was generated by Col(v)BERT$_{\text{msmarco}}$ using Google-vision.

able to place the correct page in the top five search results 1 in 10 times. Coveo performed marginally better with a score of 15.97. This poor performance should not be surprising. Traditional, keyword-based search engines can't answer questions. It's just not what they were designed to do.

4.3.2 ColBERT Baseline Performance. ColBERT outperforms our in-production search appliances in all cases. Interestingly though, its performance strongly depends on the training corpus. It varies from a mere 4.7 points better than Coveo when trained on Natural Questions (NQ) to 29.5 points better when trained on sMarco. As you'll see in the next two sections, it can be further improved upon by swapping out the underlying language model, BERT, for vBERT and performing model supervision.

4.3.3 Col(v)BERT Zero-Short Performance. Swapping out ColBERT's default LM, BERT, for our own, domain-adapted vBERT yielded across-the-board performance improvements when comparing directly by training corpus. Col(v)BERT trained on NQ beat ColBERT trained on NQ by 11 points. Col(v)BERT trained on MS MARCO beat ColBERT trained on MS MARCO by 5.4 points. And, Col(v)BERT trained on sMarco beat ColBERT trained on sMarco by 3 points. Though the delta shrank as the overall performance increased, Col(v)BERT outperformed ColBERT in all direct comparisons.

It's unsurprising that a domain-adapted LM outperformed a generic LM. The real comparison is against our in-production search appliances. The best Col(v)BERT zero-shot model (excluding Col(v)BERT based on vBERT-Large) beat Coveo by 32.42 points.

4.3.4 Supervised Col(v)BERT Performance. In all cases, the supervised model outperformed its zero-shot equivalent. The supervised NQ model is 2.3 points better than the zero-shot NQ model. The supervised MS MARCO model is 1.4 points better than the zero-shot MS MARCO model. And, the supervised sMarco model is 0.7 points better than the zero-shot sMarco model. Similar to ColBERT vs Col(v)BERT, the apparent gains from supervision decrease as the score of the zero-shot model increases; however, supervision still produced the highest-scoring model.

Because we're able to use different models when doing the zero-shot training and supervision, we were able to use vBERT-Large to train the best zero-shot model and use triples generated by that model to supervise vBERT-Base to produce the highest-scoring model overall. This final model, Col(v)BERT$_{\text{smarco-vbertlarge-googlevision}}$ in Table 3, is 47.9 points (4.0×) better than Coveo and 52.3 points (5.5×) better than Elasticsearch.

Additionally, we believe the supervised model is even better than the delta between a supervised model and the equivalent zero-shot model would lead one to believe. However, we don't have any hard data to quantify this belief. See Section 5 for an informal discussion of the difference between zero-shot and supervised models.

4.3.5 The Importance of Deduplication. A late discovery was that our original deduplication algorithm was non-optimal. ColBERT is a *passage* search engine, so we were deduplicating by *passage*. But that's not how a user expects search results to be displayed. They expect results to be deduplicated by *page*. That's the difference between Dedupe v1 and v2. Version 1 was by passage. Version 2 is by page. That's the source of the ~10 point boost in scores when we moved from v1 to v2.

4.3.6 Sentence Transformers Performance. The Sentence Transformers library contains numerous models for IR [29]. The most direct comparison to ColBERT is S-BERT trained on MS MARCO. Despite achieving similar scores on the MS MARCO reranking task,

it performed surprisingly poorly on our test set, scoring over 11 points worse than ColBERT trained on MS MARCO.

Their highest performing model, based on MPNet-Base, does outperform ColBERT and Col(v)BERT trained on MS MARCO, but under-performs Col(v)BERT-Base trained on sMarco and is significantly worse than Col(v)BERT-Base supervised by Col(v)BERT-Large at Recall@5. Their model is, not-so-usefully, the highest performing model at Recall@200 (ignoring Dedupe v2), but how many users wade through to the 20th page of search results? Thus, we feel safe claiming that our model outperforms their best semantic search model.

5 INFORMAL COMPARISON BETWEEN ZERO-SHOT AND SUPERVISED MODELS

The difference in scores on our test set between the zero-shot model, Col(v)BERT-Large trained on sMarco, and the supervised model, Col(v)BERT-Base trained on the triples generated by Col(v)BERT-Large, is only 0.16 points (dedupe v1 scores). That appears to be a negligible difference, but doesn't tell the whole story. We believe this is a limitation of our test set, and thus only have anecdotal evidence to show that the supervised model is actually *much* better than the zero-shot model.

5.1 Informal Test Corpus

Our full retrieval corpus comprises a diverse set of content sources, such as Documentation, KB Articles, Blog posts, the official VMware Glossary, TechZone, SD-WAN, and much more (as opposed to the test corpus, which was limited to Documentation and KB articles to make the comparison fair for Elasticsearch, which doesn't index other content sources). Those content sources vary widely in scale. Our documentation has >250K documents. There are more than 30K blog posts. On the opposite end of the spectrum, there are only a little over 200 pages on tanzu.vmware.com. The internal version of vAQA will also index the Machine Learning Program Office's (MLPO) Project Database, which has a mere 162 entries in it.

5.2 Informal Test Results

If one were to search for "bert" with the zero-shot model, Col(v)BERT-Large trained on sMarco, and filter the results to only show entries from the MLPO Project Database, it comes up with nothing (which means the responsive page(s) fell below ColBERT's cutoff for inclusion in the returned set of passages). If, on the other hand, one were to search with the supervised model, Col(v)BERT-Base trained with Google-vision (which only included content from our documentation, it had never seen the MLPO database content), and do the same filter, it finds 22 projects from the database related to BERT (the entry for vBERT holding the top slot).

This anecdote is not a one-off. There are many searches where, when scrolling through the list of results, the results from the supervised model just feel like better results than those from the zero-shot model. We don't have any hard numbers to back this up, but it makes intuitive sense that a supervised model trained on the retrieval corpus would produce better results than a zero-shot model that was trained on an unrelated, publicly available, benchmark corpus.

It appears to be a similar effect as that observed when domain adapting the underlying language model. vBERT outperforms BERT on VMware-specific tasks because it was trained on VMware content, so it makes sense that Col(v)BERT trained on VMware content would produce better results than Col(v)BERT trained on generic web content.

6 RETRIEVAL CORPUS RELEASE & ZERO-SHOT IR COMPETITION

In an effort to spur academic research in the field of "domain-specific information retrieval" (as a complement to all of the research into "general information retrieval", à la MS MARCO), we have published our expanded retrieval corpus, which contains the retrieval corpus used in this work plus content from additional sources, such as Blogs, Carbon Black, Tanzu, etc. We have done so as part of a Kaggle Community Competition [6], which uses the expanded retrieval corpus along with additional queries from our Google Search Console data to cover the expanded corpus. We hope this allows the research community to better refine their models by testing their ability to transfer their search capability to a domain-specific corpus. We also hope this release and competition spur other companies to release similar data sets. Then, after more data sets are available, someone can create a comprehensive benchmark, much like SuperGLUE, but for domain-specific IR.

7 FUTURE WORK

Version 1.0 of vAQA is built with English-only models, trained on English-only data sets, and indexed English-only content. Long term, we will investigate multi-lingual models and data sets [8, 31, 36], which typically lag behind state-of-the-art English models and data sets by a year or so.

Version 2.0 will explore using multiple passages as evidence for the answer [20] (as opposed to the current system which assumes the answer is contained in a single passage) and generate a single, coherent answer from the multiple evidentiary passages.

8 CONCLUSION

Traditional, keyword-based search engines served (reasonably) well for a couple of decades, but they are rapidly becoming a relic of the past. Deep language models are far more capable of understanding context and applying their knowledge to the search and question-answering domains. Using our two-stage training pipeline, where the first stage uses Col(v)BERT, powered by vBERT-Large, trained on sMarco to generate supervision training data for the second stage, Col(v)BERT, powered by vBERT-Base, produces a state-of-the-art information retrieval model that outperforms our in-production search appliances by 47.9–52.3 points (4.0–5.5×). The combination of our text preprocessor, vNLP, domain-adapted language model, vBERT, and novel supervision technique, Google-vision, significantly outperformed our in-production search appliances and public deep learning models, like Sentence Transforms.

Pairing this state-of-the-art information retrieval model with an extractive question answering model will allow us to ease the burden of anyone on the front lines of customer support by improving the search experience and directly answering customer's questions from our published content.

ACKNOWLEDGMENTS

We would like to thank Steve Liang for allowing Rick to spend so much time on this, Matei Zaharia for allowing Omar to spend so much time on this, Rob Francis for providing the Google Search Console data, the entire InstaML team for allowing us to monopolize our development server, Puneet Katyal for building the GPU cluster we used from training, Giampiero Caprino and Dimitar Shatarov for implementing JupyterLab on top of Puneet's cluster, and Steve, Giampiero, and Na Zhang for reviewing the paper.

REFERENCES

[1] Rick Battle. 2020. vBERT 2020: Release Announcement. Published-internally-on-vBetter
[2] Rick Battle. 2020. vBERT: Initial Results Pre-training BERT on VMware Content. Published-internally-on-vBetter
[3] Rick Battle. 2020. vNLP: VMware's Natural Language Preprocessor. Published-internally-on-vBetter
[4] Rick Battle. 2021. vBetter: Zero-Shot Information Retrieval. Internal-vBetter-Competition
[5] Rick Battle. 2021. Weaknesses of WordPiece Tokenization. https://medium.com/@rickbattle/weaknesses-of-wordpiece-tokenization-eb20e37fec99
[6] Rick Battle, Darien Schettler, RadhaKrishna Vuppala, and Bhavani Kumar. 2022. VMware Zero-shot Information Retrieval. http://www.kaggle.com/c/vmware-zero-shot-information-retrieval
[7] Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitinger. 2016. Research-paper recommender systems : a literature survey. *International Journal on Digital Libraries* 17, 4 (2016), 305–338. https://doi.org/10.1007/s00799-015-0156-0
[8] Luiz Henrique Bonifacio, Israel Campiotti, Vitor Jeronymo, Roberto Lotufo, and Rodrigo Nogueira. 2021. mMARCO: A Multilingual Version of the MS MARCO Passage Ranking Dataset. arXiv:2108.13897 [cs.CL]
[9] Elasticsearch B.V. 2021. Elasticsearch: The heart of the free and open Elastic Stack. https://www.elastic.co/elasticsearch/
[10] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *CoRR* abs/2003.10555 (2020). arXiv:2003.10555 https://arxiv.org/abs/2003.10555
[11] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Jimmy Lin. 2021. MS MARCO: Benchmarking Ranking Models in the Large-Data Regime. *CoRR* abs/2105.04021 (2021). arXiv:2105.04021 https://arxiv.org/abs/2105.04021
[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805
[13] Google. 2021. How Google's featured snippets work. https://support.google.com/websearch/answer/9351707
[14] Stanford NLP Group. 2021. SQuAD2.0 Leaderboard. https://rajpurkar.github.io/SQuAD-explorer/
[15] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented Language Model Pre-Training. *CoRR* abs/2002.08909 (2020). arXiv:2002.08909 https://arxiv.org/abs/2002.08909
[16] Coveo Solutions Inc. 2021. Coveo: Enterprise Search Solutions. https://www.coveo.com/en
[17] Gautier Izacard and Edouard Grave. 2020. Distilling Knowledge from Reader to Retriever for Question Answering. *CoRR* abs/2012.04584 (2020). arXiv:2012.04584 https://arxiv.org/abs/2012.04584
[18] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. *CoRR* abs/2004.04906 (2020). arXiv:2004.04906 https://arxiv.org/abs/2004.04906
[19] Daniel Khashabi, Amos Ng, Tushar Khot, Ashish Sabharwal, Hannaneh Hajishirzi, and Chris Callison-Burch. 2021. GooAQ: Open Question Answering with Diverse Answer Types. *arXiv preprint* (2021).
[20] Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Baleen: Robust Multi-Hop Reasoning at Scale via Condensed Retrieval. *CoRR* abs/2101.00436 (2021). arXiv:2101.00436 https://arxiv.org/abs/2101.00436
[21] Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Relevance-guided Supervision for OpenQA with ColBERT. arXiv:2007.00814 [cs.CL]
[22] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. arXiv:2004.12832 [cs.IR]
[23] Ankit Kumar, Piyush Makhija, and Anuj Gupta. 2020. User Generated Data: Achilles' heel of BERT. *CoRR* abs/2003.12932 (2020). arXiv:2003.12932 https://arxiv.org/abs/2003.12932
[24] 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* (2019).
[25] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *CoRR* abs/1611.09268 (2016). arXiv:1611.09268 http://arxiv.org/abs/1611.09268
[26] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *CoRR* abs/1910.10683 (2019). arXiv:1910.10683 http://arxiv.org/abs/1910.10683
[27] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. *CoRR* abs/1806.03822 (2018). arXiv:1806.03822 http://arxiv.org/abs/1806.03822
[28] Nils Reimers. 2021. Semantic Search Models. https://twitter.com/Nils_Reimers/status/1435544752938749953?s=20
[29] Nils Reimers. 2021. Sentence Transformers - Pretrained Semantic Search Models. https://www.sbert.net/docs/pretrained_models.html#semantic-search
[30] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *CoRR* abs/1908.10084 (2019). arXiv:1908.10084 http://arxiv.org/abs/1908.10084
[31] Peng Shi, Rui Zhang, He Bai, and Jimmy Lin. 2021. Cross-Lingual Training with Dense Retrieval for Document Retrieval. arXiv:2109.01628 [cs.CL]
[32] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. *CoRR* abs/2004.09297 (2020). arXiv:2004.09297 https://arxiv.org/abs/2004.09297
[33] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models. *CoRR* abs/2104.08663 (2021). arXiv:2104.08663 https://arxiv.org/abs/2104.08663
[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR* abs/1706.03762 (2017). arXiv:1706.03762 http://arxiv.org/abs/1706.03762
[35] Na Zhang and RadhaKrishna Vuppala. 2021. Information Retrieval ML Competition Solution. VMware-Internal-Confluence
[36] Xinyu Zhang, Xueguang Ma, Peng Shi, and Jimmy Lin. 2021. Mr. TyDi: A Multi-lingual Benchmark for Dense Retrieval. *CoRR* abs/2108.08787 (2021). arXiv:2108.08787 https://arxiv.org/abs/2108.08787