Systematic Error Analysis of the Stanford Question Answering Dataset

Marc-Antoine Rondeau
Microsoft Research
Montréal, Québec, Canada
marondea@microsoft.com

Timothy J. Hazen
Microsoft Research
Cambridge, Massachusetts, USA
tj.hazen@microsoft.com

Abstract

We analyzed the outputs of multiple question answering (QA) models applied to the Stanford Question Answering Dataset (SQuAD) to identify the core challenges for QA systems on this data set. Through an iterative process, challenging aspects were hypothesized through qualitative analysis of the common error cases. A classifier was then constructed to predict whether SQuAD test examples were likely to be difficult for systems to answer based on features associated with the hypothesized aspects. The classifier's performance was used to accept or reject each aspect as an indicator of difficulty. With this approach, we ensured that our hypotheses were systematically tested and not simply accepted based on our pre-existing biases. Our explanations are not accepted based on human evaluation of individual examples. This process also enabled us to identify the primary QA strategy learned by the models, i.e., systems determined the acceptable answer type for a question and then selected the acceptable answer span of that type containing the highest density of words present in the question within its local vicinity in the passage.

1 Introduction

Since the introduction of the Stanford Question Answering Dataset (SQuAD, Rajpurkar et al., 2016), research groups have directed significant efforts towards achieving a high position on the SQuAD leaderboard.[1] This competition has resulted in many new models for question answering using machine reading comprehension.

Within SQuAD, a single test example consists of three components: a question, a text passage and an answer. The answer is a span extracted from the passage answering the question. Questions were created by human annotators, who were shown a passage and asked to produce question and answer pairs. In performing the question answering task, the best performing systems employed complex attention flow mechanisms for matching questions to substrings of the text passage. These models, while varied, all belong to the same general family of neural network architectures.

In this work, we conducted a systematic error analysis on the development set of SQuAD to explain the common failures and successes of some of these models; the results can be expected to generalize to the entire family. Our goal was to explain the models' failures and successes using well defined features automatically extracted from examples. We wanted to use simple features, such as word identity, over complex features. We wanted to avoid explanations based on the human strategy used to answer a question, or complex features that cannot be extracted automatically, such as reasoning, common sense or external knowledge. Finally, we wanted to isolate a passages' readability from the strategy required to answer questions.

Our methodology used classifiers to predict the difficulty of questions. The classifier performance was used to confirm or refute the validity of a hypothesized challenge using its true and false positive rates over the entire development set. Systematic testing across all system failures and successes can reduce the risk of confirmation bias inherent to random spot checks.

A key difference with previous error analysis on SQuAD is that we looked for successes present-

[1] https://rajpurkar.github.io/SQuAD-explorer/

Proceedings of the Workshop on Machine Reading for Question Answering, pages 12–20
Melbourne, Australia, July 19, 2018.

ing the same challenges observed in failures. This confirms that the same explanations are not applicable to the successes. While many system errors could be explained in term of human challenges, features related to those challenges were usually independent of failures and successes. This can easily be missed by random spot checks relying on human evaluation.

From our evaluations, we identified a reading strategy that matched the observed failures and successes. We believe that this methodology is more robust than the common ad-hoc approaches purely based on human evaluations over a small random sample. The reading strategy we identified indicates that SQuAD is surprisingly well suited for neural network based models. While it remains a valuable resource, this now limits its suitability for further improvement of QA models.

1.1 Text Organization

In this paper, we will first present some related works, and explain how our methodology differs. A description of our methodology will follow. We will then present experimental results for three groups of hypotheses (readability, Q-words, and acceptability), and a combined model. Finally, we will describe the human analog of the models' strategy, followed by our conclusions.

2 Related Works

Sugawara et al. (2017) evaluated various datasets, in particular SQuAD, to determine how many human reading skills were required to answer questions. They described SQuAD as "difficult to read but easy to answer" for humans, finding that SQuAD requires only a few simple skills. In contrast, we are identifying skills used by machines.

FastQA (Weissenborn et al., 2017) added simple word matching features, indicating that a word was in both the passage and question, to a simple MRC model. Those simple features improved performance using this simple MRC model. We observed that variations of this feature were acceptable predictors of failures and successes

Adversarial SQuAD (Jia and Liang, 2017) added distractor sentences at the end of SQuAD examples. Model specific distractors were created by adding random words, guided by the target model's output, until it predicted a wrong answer. The resulting sentences are ungrammatical and have no semantic significance, but match words present in the question. Similarly, a more generic set of distractors was created using a simple set of rules to transform the question into a statement, and replacing keywords. The resulting sentence is grammatical and meaningful, but is irrelevant to the question. The significant number of word matches between the question and the distractor significantly reduces performance.

Those related works indicates that word to word matching, similar to the reserved engineered strategy described in Section 8, is sufficient to obtain good performance on SQuAD.

In this work, we used systematic hypothesis testing over both failures and successes to identify the strategy used by machines to reach high performance on SQuAD. Systematic testing based on automatically extracted features prevent us from relying on human explanation. It also limits confirmation bias, which is a concern for qualitative analysis. Human investigators will tend to explain errors in term of the human skills required, even when a simpler explanation is possible. It is also important to confirm that the same explanation is not applicable to the models' successes. Previous error analysis focused on errors, and ignored successes.

3 Methodology

We want to explain models' failures and successes while avoiding explanations based on human reading comprehension, and to test our explanations systematically. We defined empirical difficulty classes (Section 3.1) and used the linear separability of those classes using the extracted features to accept or reject a hypothetical explanation. We iterated qualitative analysis, hypothesis generation and the creation of corresponding feature extractors, and testing.

3.1 Difficulty Classes Used

We used the single and ensemble outputs of the three models listed in Table 1, for a total of six models. The models were chosen due to their performance on SQuAD: all were near the top of the leaderboard at the time this work began. While these models are only a subset of the models on the SQuAD leaderboard, they share similar features with the others. We believe that that our findings generalize to the others. Questions were divided into 3 classes : easy, hard and other. EASY questions were those questions where all six models re-

BiDAF	(Seo et al., 2016)
Reasonet	(Shen et al., 2017)
FusionNet	(Huang et al., 2017)

Table 1: Models used for error analysis. The single and ensemble version of each was used

Class	Count	Frequency (%)
EASY (6 EMs)	5,874	55.57
6 PMs	459	4.34
5 PMs	1,179	11.15
4 PMs	753	7.12
3 PMs	611	5.78
2 PMs	634	6.00
1 PM	631	5.97
HARD (0 PMs)	429	4.06

Table 2: Distribution of question as a function of the number of models predicting a partial match (PM). Also includes the two main classes, EASY (all models predicted exact matches, EMs), and HARD (no prediction is a match.)

turned an exact match (EM) with a human answer, and HARD questions were those where none of the answers was a partial match (PM). All other questions were placed in the OTHERS class. Table 2 shows the resulting distribution, with the OTHERS class subdivided according to the number of partial or exact matches.[2]

3.2 Classifier and Hypothesis Testing

Classifiers trained and tested on the entire development set were used to measure the linear separability of the questions' empirical difficulty class. We focused on the EASY vs (HARD ∪ OTHERS) case. Feature were accepted if the area under the receiver operating characteristic curve (AUC) was 0.6 or more; This threshold was picked based on the performance of text complexity features. Features were also accepted if they improved the AUC when combined with the existing features.

Our goal was to identify features with two properties:

1. Features are linearly correlated with failures or successes, and

2. The intersection between the feature values for question in the EASY and HARD sets is as small as possible.

[2]The 6 PMs may include up to five EMs.

This is equivalent to linear separability, which we can evaluate using classifiers.

This process allowed for hypothesis testing: a hypothesized explanation was rejected when it was not possible to create corresponding features that would improved the classifier, or be predictive by themselves.

3.3 Receiver Operating Characteristic Curves

Receiver operating characteristic (ROC) curves are used to compare the performance of classifiers. They illustrate the trade-offs between false positives and false negatives. In practice, they can measure the linear separability of the feature used in a linear classifier.

In a linear classifier, one or more features are projected down to one dimension. If the projected value is greater than the threshold t, then the example is classified as belonging to the class, otherwise it is classified as outside the class.

The ROC curve contains all the points $(P(X \geq t|c = 0), P(X \geq t|c = 1))\forall t \in \text{supp}(X)$, where X is a random variable corresponding to the projected value for a random example, and $\text{supp}(X)$ is its support. $P(X \geq t|c = 0)$ is the false positive rate, while $P(X \geq t|c = 1)$ is the true positive rate. The area under the curve (AUC) can be used to summarize the performance of the corresponding classifier. It would be 0.5 for a perfectly random classifier, and 1.0 for a perfect classifier.

3.4 Iterative Procedure

We used an iterative process where question/passage pairs were selected randomly, mainly from the HARD class described in Section 3.1. A qualitative analysis of this sample was then used to identify common features that would explain the models' failure or success. Corresponding feature extractors were then created, and used in logistic regression classifiers in order to assign questions to one of the difficulty classes described below. Good features would be correlated with either failures or successes. The explanation was then accepted if it was sufficient to separate, at least partially, the two classes, or if it improved the classifier performance when combined with previously accepted features. This process was repeated until no new hypotheses were generated.

What is the name of an algebraic structure in which addition, subtraction and multiplication are defined?
Prime numbers give rise to two more general concepts that apply to elements of any commutative ring R, **an algebraic structure where addition, subtraction and multiplication are defined**: prime elements and irreducible elements. An element p of R is called prime element if it is neither zero nor a unit (i.e., does not have a multiplicative inverse) and satisfies the following requirement: given x and y in R such that p divides the product xy, then p divides x or y. An element is irreducible if it is not a unit and cannot be written as a product of two ring elements that are not units.

Table 3: Example of hard to read passage associated with an easy to answer question

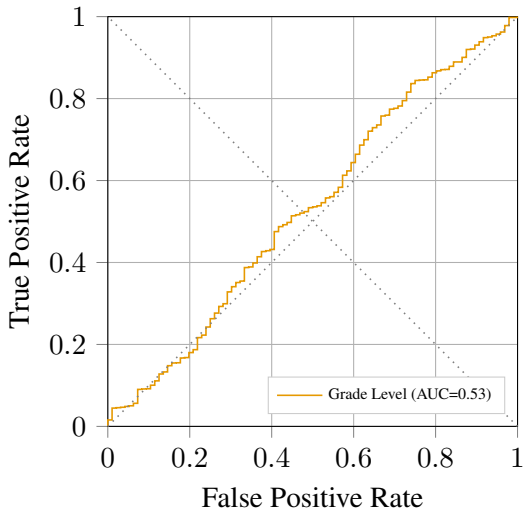

Figure 1: ROC curve using Grade level to detect EASY questions

4 Reading difficulty

Features based on text complexity and human readability were used as control. Those features were used to confirm that the hypothesized explanations were not proxies for human text complexity. They were also used to establish the performance threshold required to accept hypotheses.

We used the grade level metric (Kincaid et al., 1975), commonly used to evaluate the readability of document for humans. The grade level is a weighted sum of the average number of syllables per word and words per sentences, weighted to match reading ability expected of a student in that grade, in the US education system. Figure 1 shows the ROC curve when predicting the error class of a question given the grade level of the passage and question. The AUC is 0.53 when classifying EASY vs (HARD ∪ OTHERS), which is effectively random. Table 3 shows an example of a hard to read passage associated with an EASY question.

We also investigated other features measuring text complexity. Those features and their individual performance are described in Appendix AUsing a combination of those features, the AUC is 0.54 when classifying EASY vs (HARD ∪ OTHERS), which is effectively random. This indicates that those features are not predictors of failures or successes. In practice, the difficulty class of a question is not based on the human readability of the associated passage. In particular, complex internal dependencies and co-referral structures, which would support complicated questions, are not predictors of failures or successes. This *suggests* that the failures are not caused by the complexity of the human strategy required to solve a question, assuming complicated questions tend to be associated with complicated passages.

5 Density and Proximity to Q-Words

To shorten notation, we refer to words present in the question as Q-words. Qualitative analysis suggested that successes tended to have many Q-words nearby. Similarly, failures tended to have few Q-words in their vicinity. Table 4 shows an example of this concept extracted from SQuAD. All systems selected the same incorrect answer, which is in a region of high Q-word density. The correct human answer is in a region of lower density, and unlike the systems' answer it is not adjacent to Q-words.

Figure 2 shows the corresponding ROC curves for a classifier using density and proximity features. The number of Q-words within up to 10 words from a human answer was the best individual feature. This measures the density of Q-words in the vicinity of human answers. The AUC was 0.60 when classifying EASY vs (HARD ∪ OTHERS), 0.66 for HARD vs (EASY ∪ OTHERS), and 0.70 for EASY vs HARD.

The second best individual feature was the dis-

What was the name of the first Doctor Who story released as an LP?
The earliest **Doctor Who**-related audio release was a 21-minute narrated abridgement of the **First Doctor** television **story** *The Chase* **released** in 1966. Ten years later, the **first** original **Doctor Who** audio was **released** on **LP** record; Doctor Who and the Pescatons featuring the Fourth **Doctor**. The first commercially available audiobook was an abridged reading of the Fourth **Doctor story** State of Decay in 1981. In 1988, during a hiatus in the television show, Slipback, the first radio drama, was transmitted.

Table 4: Example of Q-word density and proximity. The systems' answer, in *italic*, is closer to Q-Words, in **bold**, than the underlined human answer.

tance between human answers and the nearest peak in the Q-word density. This measures the proximity of Q-word clusters to human answers. The AUC was 0.59 when classifying EASY vs (HARD ∪ OTHERS), 0.66 for HARD vs (EASY ∪ OTHERS), and 0.69 for EASY vs HARD.

The classifiers were able to classify EASY vs HARD more easily than EASY vs (HARD ∪ OTHERS) and HARD vs (EASY ∪ OTHERS). Those results shows that the overlap between EASY and HARD is smaller than the overlaps between EASY and OTHERS, and between HARD and OTHERS. This indicates that there is an approximate ordering HARD $\leq$ OTHERS $\leq$ EASY when going from low to high values of those density and proximity metrics.

The density and proximity features are acceptable predictors of failures and successes. This simple mechanical explanation shows that similarity between the question and the answer's surroundings will contribute to the machine difficulty of the question. The features used were based on direct, exact matches between words. Capitalization was ignored, but we did not perform any other form of normalization, or accept any other differences when matching, including trivial ones such as pluralization. This very strict matching was sufficient to create an acceptable predictor.[3]

[3]Better matching (e.g.cosine distance between word vectors) should improve performance, but would be less explainable than direct matches. As our goal is to explain errors, rather than predict them, we decided to use direct matching.

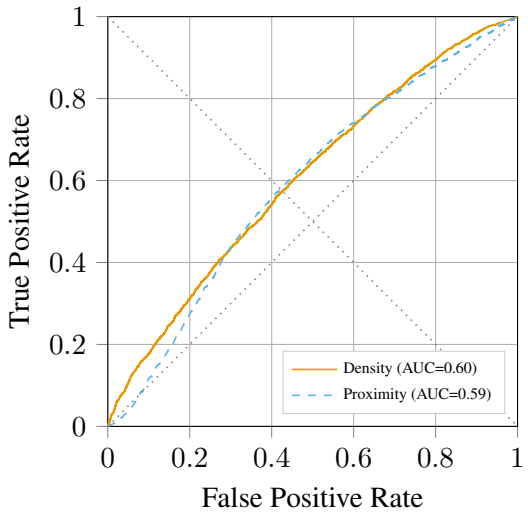

Figure 2: ROC when using Q-words density and proximity to detect EASY questions

6 Acceptability

Qualitative analysis of the false-negatives and of randomly sampled false-positives, using the density and proximity classifier, suggested that models were returning answers that were "acceptable" to the question. Questions would correspond to one or more answer types, and the models would retrieve the span belonging to one of those types with the highest Q-word density and proximity. While this notion of acceptability is hard to define, a significant portion of the answers are named entities (NEs) of various types. This can be used to test this hypothesis. We used CoreNLP (Manning et al., 2014) to identify which answers are NEs, as well as their type and competing spans.

Table 5 shows an example of acceptability. All systems select the only date presents in the passage. The correct answer is the proper name of an event. Unlike dates, "Super Bowl LI" would not usually be used to answer "when" questions, and would not generally be considered acceptable.

6.1 Typed and Competition Features

We created a typed feature indicating that a least one human answer overlapped with a NE. The AUC was 0.63 when classifying EASY vs (HARD ∪ OTHERS), 0.54 for HARD vs (EASY ∪ OTHERS), and 0.60 for EASY vs HARD. The typed feature is binary; this result is caused by the fact 53.18% of

When will Roman numerals be used again to denote the Super Bowl number?
On *June 4, 2014*, the NFL announced that the practice of branding Super Bowl games with Roman numerals, a practice established at Super Bowl V, would be temporarily suspended, and that the game would be named using Arabic numerals as Super Bowl 50 as opposed to Super Bowl L. The use of Roman numerals will be reinstated for Super Bowl LI . . .

Table 5: Example of acceptability. The systems' answer, in *italic*, is the only date in the passage. The human answer is underlined.

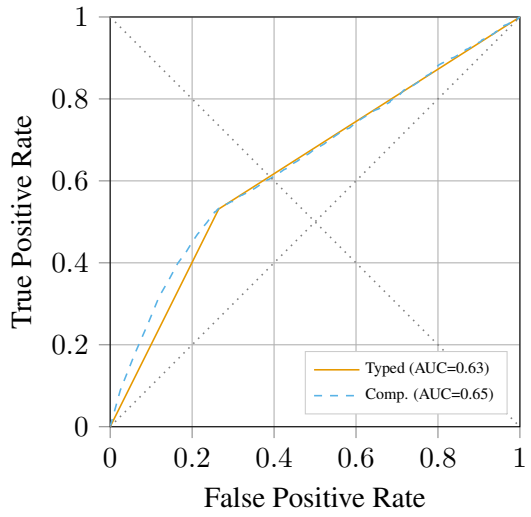

Figure 3: ROC when using typed and competition features to detect easy questions

EASY questions are typed, but only 26% of (HARD ∪ OTHERS) questions are typed. This will be investigated in more details below.

A feature indicating the acceptable spans count was also created. This feature is equal to the number of NE of the same type when least one human answer overlap with a NE, falling back to the number of words in the passage in the case where no human answer is a NE. In the NE case, this should correspond to the number of competing hypotheses; otherwise, the number of words in the passage was picked as a simple heuristic. The AUC was 0.65 when classifying EASY vs (HARD ∪ OTHERS), 0.54 for HARD vs (EASY ∪ OTHERS), and 0.61 for EASY vs HARD.

Figure 3 shows the corresponding ROC curves.

Type	Count	Freq. (%)	EASY (%)
Non-NE	6,217	58.82	44.44
All NEs	4,353	41.18	72.11
DATE	968	9.16	83.37
PER	956	9.04	69.67
LOC	632	5.98	68.99
NUMBER	618	5.85	73.46
ORG	573	5.42	63.18
Others	606	5.73	68.27

Table 6: Distribution of questions by NE type and difficulty class.

The overlap between the typed feature's predictions and the acceptable span count's prediction is clearly visible in this figure.

When used only for typed questions, the AUC was 0.59 when classifying EASY vs (HARD ∪ OTHERS), 0.62 for HARD vs (EASY ∪ OTHERS), and 0.64 for EASY vs HARD. This shows that successes and failures are correlated with the number of acceptable spans, when the answer is a NE. This will be investigated in more details below.

6.2 Named Entity Answers

Table 6 shows the distribution of question by named entity type and difficulty class. When the answer is *not* a named entity, 44.44% of questions are in the EASY class. This proportion is 72.11% when the answer is a named entity. As shown in Figure 4, the proportion of EASY questions decreases as the number of named entities of the same type in the passage increases. Figure 5 shows that the rank of the human answer, relative to the number of competing named entities, based on the combined density and proximity features, is a predictor of failures and successes. The AUC was 0.61 when classifying EASY vs (HARD ∪ OTHERS), 0.68 for HARD vs (EASY ∪ OTHERS), and 0.70 for EASY vs HARD.

Those results indicate that the models are selective and can ignore high Q-words density regions of the passage if those regions do not contain an acceptable span. They also indicates that the density and proximity of Q-words is used to select which acceptable span should be retrieved.

7 Combination of Features

Complementarity between density, proximity and acceptability, as well as some rejected features, was tested in a single combined logistic regres-

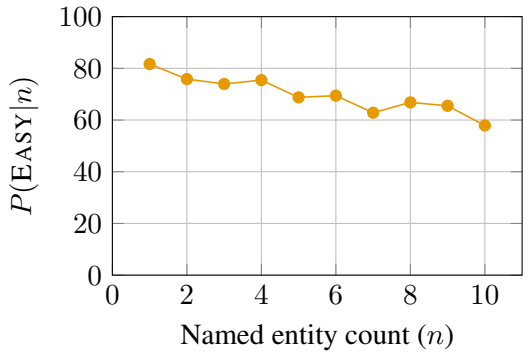

Figure 4: Proportion of EASY questions vs number of named entity per passage

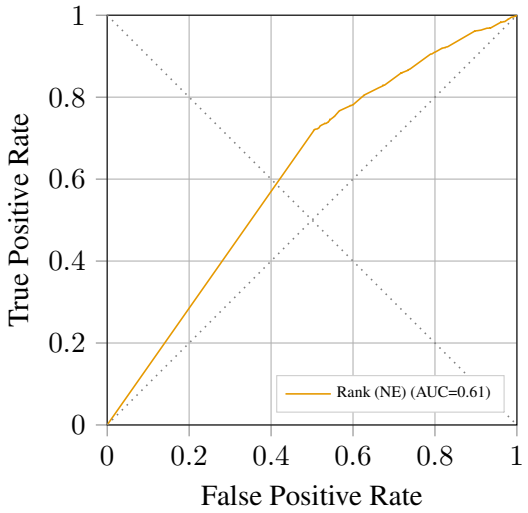

Figure 5: ROC when using the rank of named-entities human answers to detect EASY questions

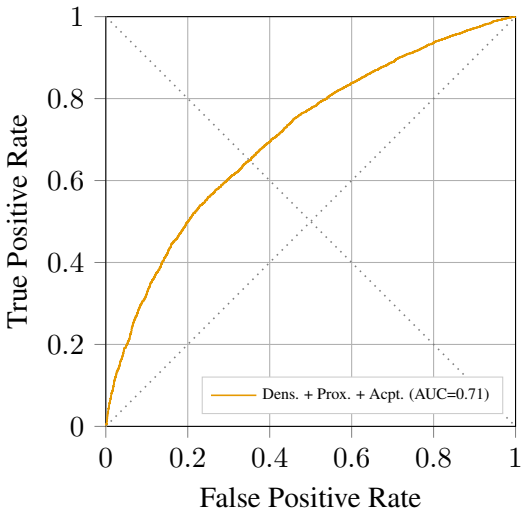

Figure 6: ROC using combination of features to detect EASY questions

Features	EASY	HARD
All Readability	0.54	0.52
All Density + All Proximity + All Acceptability	0.71	0.67
All Readability + All Density + All Proximity + All Acceptability	0.71	0.66

Table 7: Area under the curve (AUC) per group of features

sion classifier. Those features and their individual performance are described in details in Appendix B.The results are shown in Figure 6. The AUC was 0.71 when classifying EASY vs (HARD ∪ OTHERS), 0.67 for HARD vs (EASY ∪ OTHERS), and 0.74 for EASY vs HARD. Adding the readability features described in Appendix A did not significantly improve results, with AUCs of 0.71, 0.66, and 0.74 respectively. This confirms that readability is not a predictor of failures or successes, while the Q-words density and proximity, and acceptability features are.

8 Reverse Engineered Strategy

Based on the density, proximity and acceptability results presented above, we conclude that the models' QA strategy is analogous to:

1. Classify question to identify acceptable span types,
2. Extract acceptable spans from passage,
3. Rank extracted spans by Q-word density and proximity, and
4. Return best span.

This simple strategy is a human equivalent that would reproduce the failures and successes observed; it is doubtful that the models are implementing it literally. It also hides significant complexity in the acceptability and density steps, which were not properly modeled in our experiments as we wanted to ensure interpretablity. A more powerful model would be better at modeling those concepts, but such a model would be very similar to the models we are trying to analyze. This is similar to the results of the related works listed in Section 2.

9 Priming During Data Collection

We attribute the success of the simple strategy described above to priming and biases during question generation. While we cannot confirm priming experimentally, as this would involve asking a justification for the question during the initial data collection, we can extrapolate from our own attempts at question generation.

Reading the passage will prime the question creators towards questions based on interrogative paraphrases of the passage. As noted by Sugawara et al. (2017), "SQuAD was difficult to read," which should further magnify this effect: when the passage is hard to read, it is easier and faster to scan it for a sentence stating a fact and to reformulate that sentence as a question. In particular, since crowdworkers are not motivated by a genuine need for information, we can expect them to use the first question that came to mind. Table 3 shows such an example, where the question is a slight reformulation of part of the passage.

We find this priming issue concerning, and suspect that it affects many datasets. It should be possible to avoid it by using true questions, collected from various sources. Those questions should be the product of a genuine need for information rather than created for the sake of creating a question, and should be created before reading a potentially answering passage. Those can be matched to relevant documents, and answered by human annotators. If keyword search is used to retrieve relevant documents, there is of course a risk of retrieving documents containing declarative paraphrases of the questions, which would effectively prime the passage on the question.

10 Conclusion

We presented a methodology used to systematically analyze the errors of six models on SQuAD. This methodology relies on simple feature extractors and classifiers to ensure that any hypothesized explanation does not also co-occur with correct answers. By iteratively sampling falsely negative and positive predictions of this classifiers, we were able to reverse engineer a simple QA strategy that would match the models' failures and successes. While labor intensive[4], this methodology avoids confirmation bias during a qualitative analysis of a random sample. In particular, human readability might mask the true cause of an error, as human investigators will tend to explain errors by the challenges they faced when examining the question. This methodology can be applied to large datasets, and also ensures that the errors are attributed to well defined causes. We recommend its use when the challenges remaining in a dataset need to be identified.

We attribute the success of the simple strategy we identified to priming by the passage during question generation. This limits the challenges, for machines, truly present in SQuAD, and indicates that, while necessary, good performance on SQuAD is not sufficient to say that a machine reading question answering model would have good performance in general. As such, we recommend the use of datasets where question creation is independent of the passage, such as:

- MSMarco (Nguyen et al., 2016)
- NarrativeQA (Kociský et al., 2017)
- NewsQA (Trischler et al., 2017)
- SearchQA (Dunn et al., 2017)
- TriviaQA (Joshi et al., 2017)

[4] Approximately 3-4 weeks, mostly creating and testing hypotheses.

Acknowledgements

We would like to thank Eric Lin, Peter Potash, Yadollah Yaghoobzadeh, and Kaheer Suleman for their feedback and helpful comments. We also thanks the anonymous reviewers for their comments.

References

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *CoRR*, abs/1704.05179.

Hsin-Yuan Huang, Chenguang Zhu, Yelong Shen, and Weizhu Chen. 2017. Fusionnet: Fusing via fully-aware attention with application to machine comprehension. *arXiv preprint arXiv:1711.07341*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly

supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1601–1611.

Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. *Chief of Naval Technical Training, Research Branch Report 8-75*.

Tomás Kociský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2017. The narrativeqa reading comprehension challenge. *CoRR*, abs/1712.07040.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

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. *arXiv preprint arXiv:1611.09268*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2017. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1047–1055. ACM.

Saku Sugawara, Yusuke Kido, Hikaru Yokono, and Akiko Aizawa. 2017. Evaluation metrics for machine reading comprehension: Prerequisite skills and readability. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 806–817.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200.

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Fastqa: A simple and efficient neural architecture for question answering. *arXiv preprint arXiv:1703.04816*.