Evaluating Surprise Adequacy for Question Answering

Seah Kim
School of Computing, KAIST
Daejon, Republic of Korea
bbaa2837@kaist.ac.kr

Shin Yoo
School of Computing, KAIST
Daejon, Republic of Korea
shin.yoo@kaist.ac.kr

ABSTRACT

With the wide and rapid adoption of Deep Neural Networks (DNNs) in various domains, an urgent need to validate their behaviour has risen, resulting in various test adequacy metrics for DNNs. One of the metrics, Surprise Adequacy (SA), aims to measure how surprising a new input is based on the similarity to the data used for training. While SA has been evaluated to be effective for image classifiers based on Convolutional Neural Networks (CNNs), it has not been studied for the Natural Language Processing (NLP) domain. This paper applies SA to NLP, in particular to the question answering task: the aim is to investigate whether SA correlates well with the correctness of answers. An empirical evaluation using the widely used Stanford Question Answering Dataset (SQuAD) shows that SA can work well as a test adequacy metric for the question answering task.

KEYWORDS

Deep Learning, Natural Language Processing, Software Testing

1 INTRODUCTION

Deep Learning (DL) and Deep Neural Networks (DNNs) have been rapidly being adopted in various domains [7]. One of the most successful tasks for DL has been image recognition [4, 17], which quickly found applications such as autonomous driving [2] and medical imaging [10]. Due to the safety critical nature of these applications, testing and validation of DNNs have received much attention recently [5, 11, 13, 18].

However, most of the existing work on testing of DL systems have focused on image recognition tasks, especially image classification. For example, almost all test adequacy metrics that have been proposed recently, such as Neuron Coverage [13], Strong Neuron Activation Coverage (SNAC) [11], and Surprise Adequacy (SA) [5], have all been primarily evaluated against image classifiers, using benchmarks including MNIST [8] and CIFAR10 [6]. We suspect there are multiple reasons for this. Besides the fact that its application areas are the most immediate safety critical ones, the image recognition domain is intuitive to understand, and mature enough to possess multiple widely studied benchmarks, resulting in many new techniques first being evaluated in the domain.

In an attempt to expand the scope of DL testing techniques, this paper applies one of the latest test adequacy criteria, Surprise Adequacy (SA) [5], to the domain of Machine Comprehension (MC) and Question Answering (QA). MC/QA is formulated as answering a natural language query about a given context paragraph. Helpful test input for an MC/QA model will be a pair of context paragraph and a valid query (i.e., a query that is genuinely about some information in the context paragraph) that the model cannot answer properly. From the SA perspective, we would assume that, if the pair of a paragraph and a query is similar to ones observed during training, the model is more likely to answer correctly. We would also assume that, if the pair is not familiar, i.e., surprising, the model is more likely to answer incorrectly. The capability to make this prediction is important for testing of DL systems because almost always the only available test oracle is human labelling. To evaluate the model performance against new pairs of a context paragraph and a query, a human must read both the paragraphs and queries and decide whether the answer is correct, incurring a huge cost. A reliable prediction of model performance can allow us to prioritise inputs that are likely to reveal incorrect behaviours.

While we expect the basic principles behind SA to hold for the MC/QA task, the natural language input format entails a couple of important differences from image recognition tasks. First, words and sentences are discrete entities, compared to images of real-world objects that show more continuous characteristics (e.g., colours inhabit a continuous domain, whereas words are discrete). Whether input similarity is still meaningful in the discrete input space remains to be confirmed. Second, natural language inputs tend to be of variable sizes, whereas images are easily rescaled into fixed sizes. The variability makes it difficult to capture the internal states of DNN models in fixed shapes.

We evaluate the feasibility of applying SA to MC/QA tasks despite the differences from image recognition tasks, using a widely studied MC/QA benchmark with over 100K queries, SQuAD [15], and a Bi-directional Attention Flow (BiDAF) question answering model [16]. The results show that SA is indeed correlated with the correctness of model behaviour, allowing us to prioritise unseen test inputs for manual labelling according to the likelihood of model producing incorrect answers. Using the SA values and labels about the correctness of the answer, both obtained from the original training data of the MC/QA model, we can choose a subset of inputs that lead to 17.6 percent point decrease in model performance (measured in Exact Match score) when compared to random sampling.

The rest of this paper is organised as follows. Section 2 describes the background of test adequacy for DNNs, including a detailed description of SA. Section 3 presents the Machine Comprehension/Question Answering model that we study. Section 4 presents the research questions, and Section 5 describes our experimental setup. Section 6 presents and discusses the results of the empirical evaluation. Section 7 goes through the threats to validity, and Section 8 concludes with an enumeration of future work.

2 BACKGROUND

This section contains background information about existing work on testing of DL systems, as well as a detailed description of Surprise Adequacy that we use in this paper.

2.1 Test Adequacy for Deep Neural Nets

In traditional software testing, the ideal test input is one that will reveal faults in the target software. However, directly obtaining such ideal inputs is not possible, as the only way of ensuring that we have all fault revealing input is to perform exhaustive testing that takes an infeasible amount of time for any practical software system. Consequently, we define surrogate measures that are thought to be correlated with fault revealing capability, such as code coverage [1] and capture how well an input achieves such surrogate measures in the form of test adequacy.

In testing of DL systems, we can similarly define test adequacy of an input as the likelihood of the input revealing suboptimal behaviour of the DL model under testing. Early test adequacy metrics for DL systems tended to extend the coverage analogy by defining criteria for elements in certain sets to be covered. For example, Neuron Coverage (NC) considers a neuron in a DNN *covered* by a test suite (i.e., a set of test inputs) if, during the execution of the test suite, the neuron is activated above a predefined threshold [13]. The intuition is that a higher NC means the underlying DNN has been executed by a more diverse set of inputs, which in turn has a higher chance of revealing undesired behaviour compared to a less diverse set. Similarly, k Multisection Neuron Coverage (kMNC) defines k different buckets over the range of neuron activation values for all neurons and computes the ratio of covered buckets to the number of all buckets [11].

2.2 Surprise Adequacy

Surprise Adequacy (SA) has been proposed by Kim et al. to overcome the weaknesses of previous test adequacy criteria [5]. The weaknesses are twofold. First, some of the proposed test adequacy criteria tend to saturate too easily, losing the ability to differentiate test suites too quickly [11]. Second, Kim et al. point out that criteria such as NC and kMNC tend to produce the same or very similar value to different individual test inputs, losing the ability to differentiate individual test inputs [5].

Going beyond the coverage analogy, Kim et al. defines the test adequacy of an input as the similarity to the data observed during the training of the target DNN. The intuition is that if an unseen new input is nevertheless similar to ones observed during training, the model is likely to behave correctly, whereas if an unseen new input is significantly different from ones observed during training, the model is much less likely to behave correctly. The similarity to the training data is observed by collecting and analysing Activation Traces (ATs).

2.2.1 Activation Trace. AT is what corresponds to (partial) execution traces in traditional software. Given a DNN, Kim et al. defines an AT vector to be the dump of a whole (or partial) neural layer during its execution by an input. It corresponds to the state of the DNN model, captured as a datapoint in the latent feature space used by the model. By collecting ATs from all inputs in the training data, we can store the internal representation of the training data.

2.2.2 Likelihood based SA. Given the set of AT vectors obtained from the training data, $\alpha_i^T \in T$, and an AT vector from a new and unseen input, α_n, Kim et al. proposes Likelihood based Surprise Adequacy (LSA) as follows. First, perform Kernel Density Estimation over $\alpha_i^T \in T$, with a Gaussian kernel function K. The density function is then given as $\hat{f}(\alpha_n) = \frac{1}{|T|} \sum_{\alpha_i^T \in T} K(\alpha_n - \alpha_i^T)$. The LSA of the new activation vector α_n is computed as the negative log of the density: $LSA(\alpha_n) = -\log \hat{f}(\alpha_n)$: the lower the probability density of an activation trace vector of a new input is, the more surprising that input is to the model.

2.2.3 Mahalanobis Distance based SA. We propose Mahalanobis distance SA (MDSA) as an extension to the original Distance based SA (DSA) proposed by Kim et al. [5]. MDSA is inspired by the use of Mahalanobis distance by Lee et al. [9] in an attempt to detect out of distribution inputs, which is essentially a goal shared by the original SA. Mahalanobis Distance measures the distance between a probability distribution and a single data point [12]. Given a vector $\vec{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ and a set of observations with mean values $\vec{\mu} = (\mu_1, \mu_2, \ldots, \mu_n)$ and a covariance matrix C, the distance D_M between $\vec{\alpha}$ and $\vec{\mu}$ is defined as $D_M(\vec{\alpha}, \vec{\mu}) = \sqrt{(\vec{\alpha} - \vec{\mu})^T C^{-1} (\vec{\alpha} - \vec{\mu})}$. We simply compute the Mahalanobis Distance between the activation vector of a new unseen input and the distribution of activation vectors obtained from the training data. The farther away an activation trace vector of a new input is from the training data distribution, the more surprising the input is to the model.

3 MACHINE COMPREHENSION/QUESTION ANSWERING MODEL

We use the Bi-directional Attention Flow (BiDAF) model [16] for our empirical evaluation. While BiDAF is not the state-of-the-art model for the SQuAD dataset (see Section 5.1), it has subsequently inspired cutting-edge models like BERT [3] and ELMo [14]. We choose BiDAF because its bi-directional attention flow layer provides a single representation that reflects both the context and query words. Here, we briefly describe the core mechanism of the BiDAF model that is relevant to how we extract ATs: for more technical details, please see the original publication [16].

3.1 Bi-directional Attention Flow Model

The BiDAF model consists of six layers. Suppose $\{x_1, \ldots, x_T\}$ and $\{q_1, \ldots, q_J\}$ represent the words in the context paragraph and the query respectively. The lowest two layers are embeddings for characters and words, concatenated and processed into d-dimensional vectors, resulting in $X \in \mathbb{R}^{d \times T}$ and $Q \in \mathbb{R}^{d \times J}$. The third layer is a contextual embedding layer, which learns the interactions between words using a bi-directional LSTM, resulting in $H \in \mathbb{R}^{2d \times T}$ from X and $U \in \mathbb{R}^{2d \times J}$ from Q.

The fourth layer is the attention flow layer, which is the core of the BiDAF model. First, we construct $S \in \mathbb{R}^{T \times J}$, where S_{tj} indicates the similarity between the t-th context word and the j-th query word. Subsequently, we compute two sets of attention weights.

- Context-to-query Attention ($\tilde{U} \in \mathbb{R}^{2d \times T}$): this signifies how close all the query words are to each context word, hence the size of $2d \times T$.
- Query-to-context Attention ($\tilde{H} \in \mathbb{R}^{2d \times T}$): this signifies how close to one of the query words each context word is, hence the size of $2d \times T$.

Since BiDAF answers the query using a phrase in the context paragraph, the attention flow is aligned with the length of the context paragraph. Once the bi-directional attention matrices $\tilde{U}$ and $\tilde{H}$ are computed, this layer returns the following concatenation as the output: $[h; \tilde{u}; h \circ \tilde{u}; h \circ \tilde{h}]$, where $[;]$ is vector concatenation across rows, and $\circ$ is element-wise multiplication. Each context word is represented as a query-aware encoding of length $8d$, as each of h, $\tilde{u}$, $h \circ \tilde{u}$, and $h \circ \tilde{h}$ is of length $2d$. We hereafter refer to each of these four $2d$ vectors as a *section* of the attention flow layer.

3.2 Extraction of Activation Traces

We use the output of the attention flow layer, i.e., the query-aware encoding of all context paragraph words, as the activation trace. Given a question answer input with T words in the context paragraph, we get T vectors, each of length $8d$ (or $2d$, if using individual sections). However, it is not possible to aggregate ATs from different inputs, as the number of words in the context paragraph can differ across inputs, making it impossible to perform KDE or compute Mahalanobis Distance. Clipping T to a fixed length is also not ideal, as the correct answer phrase may be clipped from the context paragraph as a result.

This is one of the major differences between image-based DNNs and natural language processing DNNs: unlike images that can be rescaled to a fixed size, natural language inputs are inherently variable lengths, making it difficult to capture the internal state of DNNs in a fixed shape. For this study, we take the average of all T query-aware context word representations, resulting in an AT vector of dimensionality of $8d$ (or $2d$) per question answering input. While the averaging definitely results in information loss, we take the simplest approach to evaluate the feasibility of applying SA to variable length natural language inputs. We leave better handling of variable length inputs as future work.

4 RESEARCH QUESTIONS

Existing literature has evaluated SA mostly with image-based tasks. We aim to answer the following research questions to evaluate SA against Question Answering task in NLP.

RQ1. Representativeness of AT: We investigate whether ATs extracted from the BiDAF model can effectively capture the internal state of the model, especially despite the loss of information due to the averaging of context word vectors. We posit that, if the extracted ATs capture the internal behaviour of the BiDAF model, we should be able to see some semantic patterns among the extracted ATs. We visualise the distribution of ATs and perform a qualitative analysis based on the types of queries to answer RQ1.

RQ2. Effectiveness of SA: We investigate whether SA is correlated with the correctness of the produced answers. We analyse the ATs extracted from the training data, and compute the SA values of each test input to study the correlation between SA and the correctness of the answer. To answer RQ2, we perform a statistical hypothesis test to compare SA values from both correctly and incorrectly answered queries. We also sort queries according to their SA values, and see if higher SA values result in more incorrect answers.

RQ3. Differences among SA types: We study whether one type of SA metric is better than others. We compute and compare two different SA values, LSA and MDSA, for every input in the data. In addition, we consider four additional variations of LSA, each based on a section of the output of the attention flow layer. We compared the effectiveness of each type of SA to answer RQ3.

5 EXPERIMENTAL SETUP

5.1 Dataset

We use version 1.1 of the Standford Question Answering Dataset (SQuAD) [15], which is a widely used dataset for the evaluation of Machine Comprehension and Question Answering task. SQuAD consists of more than 100,000 crowd-sourced questions extracted from Wikipedia articles. A given context paragraph corresponds to one paragraph of an article, and answers for each question are guaranteed to be a segment of a context paragraph. There are a total of 87,599 questions in training data and 10,570 questions in test data. We use the 87,599 questions to obtain the distribution of the training data, which is used to compute SA values for each of the 10,570 questions in the test data.

5.2 Model

For evaluation, we used pre-trained weights (`bidaf_50.h5`) of the `keras` implementation of BiDAF, available from https://github.com/ParikhKadam/bidaf-keras.[1] The pre-trained model was trained with SQuAD v1.1 using batch size of 16 and 50 epochs. The embedding dimension, d, was set to 400. There was no limit on the length of both passage and question. The training and test set accuracies of the pre-trained model are 51.86% and 44.97% respectively.

5.3 Configurations for SA

Kim et al. suggest removing elements of AT vectors that show low variance to reduce the computational cost of KDE [5]. Since each section of our AT has different ranges of activation values, we set up different thresholds by sections in our evaluation: we use 10^{-2} for LSA based on section 1 and 2 (h and $\tilde{u}$, respectively), and 10^{-4} for LSA based on section 3 and 4 ($h \circ \tilde{u}$ and $h \circ \tilde{h}$, respectively), as well as all sections. We compute KDE using `scikit-learn` version 0.22, and implement Mahalanobis Distance using `numpy` version 1.14.5. All experiments have been run using Python version 3.6 on a machine equipped with Intel i7-8700 CPU with 32GB RAM running Linux 16.04 LTS.

6 RESULTS

This section presents the results of the empirical evaluation and answers three research questions.

6.1 Representativeness of AT (RQ1)

RQ1 concerns the representativeness of ATs, i.e., whether ATs we extract actually capture the internal latent features of our model. For qualitative analysis of ATs, we visualise the AT vectors extracted from the questions in the test data using all four sections,

[1] We use the `keras` version because the pre-trained weights for the original TensorFlow implementation is not publicly available: see https://github.com/allenai/bi-att-flow/issues/107

based on the interrogatives of the questions, e.g., “When”, “Where”, etc. For visualisation of the distribution of ATs, we perform PCA to reduce the dimensions to 50 and subsequently use t-SNE to represent reduced AT vectors in two-dimensional space (133 yes/no questions conjugated with “be” or “do” have been excluded). Figure 1 shows the distribution of ATs: different interrogatives are colored respectively.

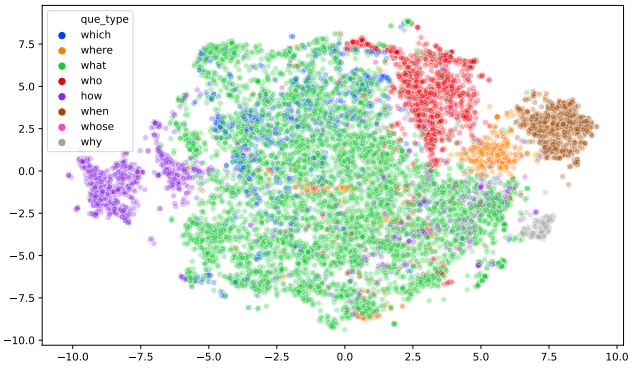
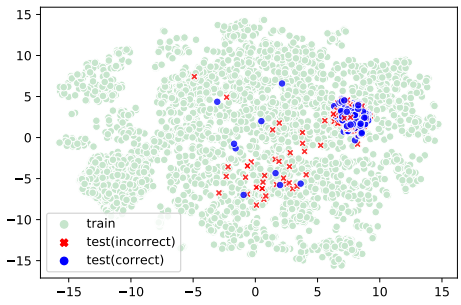


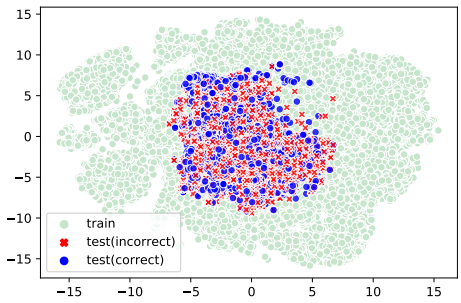
Figure 1: AT visualisation by question words using PCA to 50 dimensions followed by t-SNE

Despite the loss of information due to the averaging the word vectors (see Section 3.2), the ATs tend to show sufficient separation between interrogative types. “When (brown)”, “How (purple)”, “Who (red)”, “Why (gray)”, and “Where (orange)” occupy different AT subspaces and are clustered together. “What (green)” is the most widely spread, which may be partially explained by the various types of questions that start with “what”, such as “What time was...” or “What place was...”. Largely overlapping clusters also make intuitive sense: “which (blue)” and “whose (pink)” ATs are overlapping with “what” and “who”, respectively.

We also investigate whether ATs that correspond to correctly and incorrectly answered questions are clearly separated as well. Figure 2a and 2b show “When” and “What” ATs only, with different colors indicating correct and incorrect answers. We also plot the ATs of the training data for these interrogatives. The plots show mixed results. The correct answers for the “when” questions tend to cluster together, suggesting the existence of a familiar (i.e., unsurprising) region of input. However, the training ATs overlap with almost all test ATs, which is the same pattern observed with “what” ATs. We suspect two potential reasons for the lack of separation. First, the training accuracy of the pre-trained model is in itself not very high (accuracy of 51.86). meaning that the locations of the training ATs may not represent the region of inputs for which the model performs well. Second, two dimensions may be too low to show the separation between correct and incorrect answers. **Answer to RQ1:** based on these results, we argue that there is sufficient evidence suggesting that ATs do capture the internal behaviour of the model in a meaningful way.



(a) When



(b) What

Figure 2: AT visualisation with answer correctness

6.2 Effectiveness of SA (RQ2)

RQ2 concerns the correlation between SA and the correctness of the answer. To investigate this, we first perform a one-sided Mann-Whitney U test between LSA values computed from ATs of correct and incorrect answers respectively, using all four sections (LSA_A). The null hypothesis is that there is no difference between LSA_A values of correct and incorrect answers, whereas the alternative hypothesis is that correct answers have lower LSA_A values than incorrect answers.

Table 2: LSA_A Comparison

LSA_A	mean	std
Correct	-158.365	23.864
Incorrect	-152.786	23.304
p-value	4.327e-33	

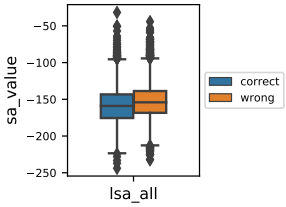


Figure 3: Boxplot of LSA_A

Table 2 shows the descriptive statistics of LSA_A, as well as the p-value from the Mann-Whitney U test, which shows a statistically significant difference. However, the boxplots in Figure 3 show that there is a significant overlap between these two values, potentially reflecting the low accuracy of the pre-trained model.

However, LSA_A can still effectively prioritise incorrect answers, as can be shown in Figure 4. We sort all test inputs according to

Table 1: Comparison of MDSA, LSA_A, LSA_1, LSA_2, LSA_3, and LSA_4 values. Apart from LSA_4, all SA values are smaller for inputs that lead to correct answers, as can be seen from the p-values of Mann-Whitney U Test.

	MDSA		LSA_A		LSA_1		LSA_2		LSA_3		LSA_4	
	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
mean	3722.957	3970.948	-158.365	-152.786	-27.163	-26.648	-125.975	-120.421	194.075	-192.243	-177.154	-175.968
std.	2137.343	2820.148	23.864	23.304	7.776	7.897	20.492	20.145	14.882	15.831	19.364	21.287
p-value	7.5879e-08		4.327e-33		4.983e-05		6.504e-43		1.871e-11		0.022	

their LSA_A values, in ascending and descending order respectively. Subsequently, we take first 100, 250, 500, 750, 1,000, and multiples of 1,000 inputs following the sorted order, and evaluate the performance of the model over these sets of inputs, recording the Exact Match (EM), which is the percentage of questions that have exact correct answers, and F1 score, which is computed from comparing the words in the produced answer to those in the ground truth answer. Inputs with higher LSA_A lead to poor performance, and lower LSA_A to high performance, a trend that is shared between both metrics. **Answer to RQ2:** This trend suggests that it is possible to prioritise incorrect answers for human labelling based on SA.

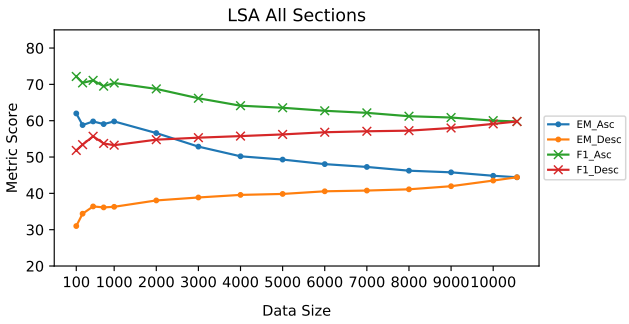


Figure 4: Model Evaluation on selected test data by sorted LSA_A values

6.3 Differences Among SA Types (RQ3)

Finally, we compare different types of SA values. Table 1 contains descriptive statistics of all SA values studied: MDSA and LSA_A, as well LSA_1, LSA_2, LSA_3, and LSA_4 that are computed from four sections of the attention flow layer output respectively. We perform the same Mann-Whitney U test between SA values of correct and incorrect answers: all types of SA except LSA_4 show that SA values from the correct inputs are statistically significantly smaller than those from the incorrect inputs.

However, for the purpose of prioritisation, we are more interested in the extreme SA values, rather than their overall distributions. To investigate this, we compare the EM score of inputs with the highest top 100 SA values to the average EM score of randomly sampled 100 inputs (repeated 20 times): the results are shown in Table 3. All SA types produce sets of inputs with significantly low EM scores when compared to random sampling, suggesting that they all can be used to effectively prioritise inputs. Figure 5 also show the same prioritisation results as in Figure 4, with the same trend. **Answer to RQ3:** while the studied SA types show different distributions, they all can be used to effectively select inputs that lead to poor model performance.

Table 3: EM Scores of Top 100 SA Inputs vs. Avarege EM Scores of Random 100 Inputs (Sampled 20 times)

SA Type	Top 100	Random 100
MDSA	27.0	44.6
LSA_A	31.0	45.35
LSA_1	44.0	44.35
LSA_2	30.0	45.95
LSA_3	38.0	42.2
LSA_4	30.0	44.95

7 THREATS TO VALIDITY

Threats to internal validity concerns factors that may influence the observed effects. In our study, this includes the validity of BiDAF model itself, as well as tools that we use to extract and analyze ATs and SA values. To address concerns, we have used the publicly available implementation of the BiDAF model, and directly adopted the pre-trained model weights. Our analysis has been performed using widely studied statistical packages, such as `scipy` and `scikit-learn`. Threats to external validity concern any factors that may prevent generalisation of our observation. This work is an early feasibility study for the application of SA to DNNs with language inputs, and consequently, the scope of the study is limited. Only additional future work with more models can ensure generalisability. Finally, threats to construct validity concerns situations where used metrics may not reflect the abstract properties they claim to measure. We use EM and F1 scores to evaluate our claims about SA: both are intuitive evaluation metrics that are widely used by many machine learning literature, and there is little room for misunderstanding.

8 CONCLUSION AND FUTURE WORK

We conduct a feasibility study for applying Surprise Adequacy (SA) metric to QA models in the Machine Comprehension domain. SA has been mainly evaluated with image recognition models; this study aims to extend the scope of its applicability. Since the cost of human labelling in text-based Machine Comprehension is significantly high, being able to select inputs that are likely to reveal incorrect behaviour can help to prioritise the human effort for labelling more efficiently. An empirical evaluation of applying SA to the Bi-directional Attention Flow model, using a widely studied

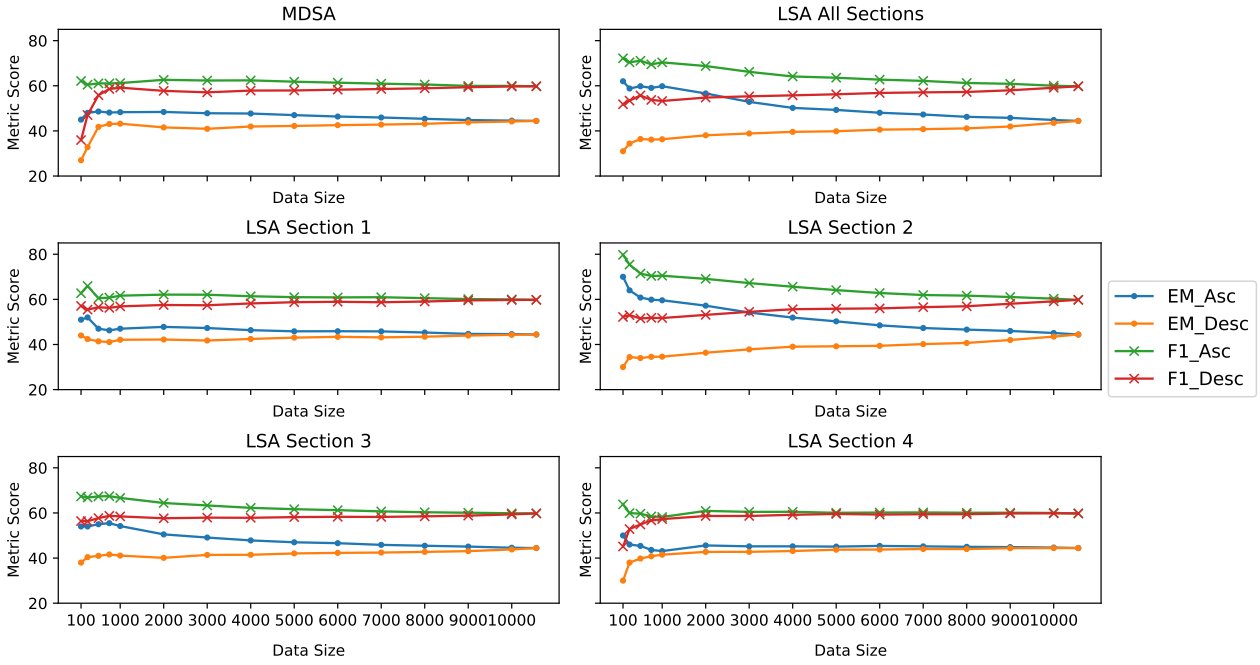


Figure 5: Model Evaluation on selected test data by sorted SA values(Ascending/Descending)

Question Answering benchmark, SQuAD, shows that, despite some challenges, SA can effectively prioritise inputs that lead to low model performance. Future work includes evaluation using more diverse QA models, as well as improved AT extraction techniques that can handle variable length model inputs more appropriately.

ACKNOWLEDGEMENT

This work was supported by the Engineering Research Center Program through the National Research Foundation of Korea funded by the Korean Government (MSIT) (NRF-2018R1A5A1059921), Institute for Information & communications Technology Promotion grant funded by the Korean government (MSIT) (No.1711073912), and the Next-Generation Information Computing Development Program through the National Research Foundation of Korea funded by the Korean government (MSIT) (2017M3C4A7068179).

REFERENCES

[1] Paul Ammann and Jeff Offutt. 2008. *Introduction to Software Testing* (1 ed.). Cambridge University Press, New York, NY, USA.

[2] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. 2015. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*. 2722–2730.

[3] 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

[4] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. 2013. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1915–1929.

[5] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding Deep Learning System Testing using Surprise Adequacy. In *Proceedings of the 41th International Conference on Software Engineering (ICSE 2019)*. IEEE Press, 1039–1049.

[6] Alex Krizhevsky. 2009. *Learning Multiple Layers of Features from Tiny Images*. Technical Report. University of Toronto.

[7] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436.

[8] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist* (2010).

[9] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. In *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 7167–7177.

[10] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, and Clara I. Sánchez. 2017. A survey on deep learning in medical image analysis. *Medical Image Analysis* 42 (2017), 60 – 88.

[11] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepGauge: Multi-granularity Testing Criteria for Deep Learning Systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (ASE 2018)*. ACM, New York, NY, USA, 120–131.

[12] Prasanta Chandra Mahalanobis. 2018. Reprint of: Mahalanobis, P.C. (1936) "On the Generalised Distance in Statistics.". *Sankhya A* 80, 1 (2018), 1–7.

[13] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2017. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17)*. ACM, New York, NY, USA, 1–18.

[14] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR* abs/1802.05365 (2018). arXiv:1802.05365

[15] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. *CoRR* abs/1606.05250 (2016). arXiv:1606.05250 http://arxiv.org/abs/1606.05250

[16] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *CoRR* abs/1611.01603 (2016). arXiv:1611.01603

[17] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[18] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, 303–314.