Challenging Semantic Role Labels

Martin Carrasco
VU Amsterdam
m.carrasco.castaneda@students.vu.nl

1 Introduction

Testing whether a machine learning model accurately represents phenomena is an issue in different areas of machine learning (Das and Rad, 2020). A good machine learning model is an accurate representation of phenomena. Developing such a system presents difficulties when evaluating what *good* constitutes. During training, the training data is expected to serve as a model of the world such that a machine learning model can learn general patterns and rules that will generalize well. However, superficial or non-generalizable characteristics may be acquired, generating a bias in the trained model. The evaluation of a model is done using a test dataset that remains untouched until the final evaluation is performed, thus ensuring a sterile test environment with what is assumed to be a representative sample of a state of the world. The tests, however, can inadvertently fail in representing the phenomena that the model should learn (Rajpurkar et al., 2018). Behavior testing is one of the ways a system can be tested for its actual predictive reach. Among the plethora of methods for testing systems, behavior testing is an approach from software engineering. It focuses on inputs and outputs while keeping the model as a black box (Beizer, 1995). *Checklist* is a tool for behavioral testing on NLP models (Ribeiro et al., 2020a). This framework presents interesting testing capabilities in three tasks:

1. **Sentiment Analysis**: Predicting the sentiment of a given sentence (*positive* or *negative*)

2. **Duplicate Question Detection**: Detecting whether 2 questions are semantically equivalent (Wang et al., 2018)

3. **Machine Comprehension**: Reading text and answering questions about it (Rajpurkar et al., 2016)

The authors use *failure rate* to evaluate how good a model performs on a set of *CheckList* tests. It is the percentage of tailor-made tests that the ML model fails. The metric is calculated on a given task for a given set of tests. A *CheckList* matrix is a collection of tests, where each row is a *capability* (such as NER, SRL), and each column is a test type (such as INV, MFT, or DIR). MFTs (Minimum Functionality Tests) are concerned with the evaluation of a specific case where we know which label should be assigned. The expected label is compared to the predicted, and that concludes the test. In INV (invariance) tests, a characteristic is usually added, altered, or *perturbed*. This newly altered sentence is compared to the base case under the assumption that the classification should not change. DIR (Directional) tests are also related to alterations, but a particular shift in the probability of the label is expected, either up or down.

Different methods can be used to generate particular tests (or fill out cells, as the authors call them). Perturbing an existing dataset, creating manually curated samples, or making use of *CheckList* perturbation templates are all valid choices. The first two methods are self-explanatory, while the perturbation templates require further explanation. Templates can be created with special sentences that generate instances of the given word. For example, *"The {NOUN} {VERB}"* would generate all combinations in the set of NOUN and VERB.

Additionally, *CheckList* provides a masked-model approach, where a RoBERTa model generates predictions when the {mask} tag is used. For example, in *"The child {VERB}" often*, it would generate possible suggestions to be filtered if the user wants to divide them into different categories. To better illustrate how *CheckList* works we present an example provided by the authors on the *Sentiment* (sentiment analysis) capability (row) using the *Minimum Functionality Test* (column) on a particular sentence (cell): *"The company is Aus-*

tralian" has actual label of **neutral** sentiment and can be created with the template *"The company is {country}*. On this particular test, the authors report the failure rates as follows: RoBERTa has a failure rate of 81.8% and BERT of 96.4% while Google's, Amazon's, and Microsoft's models have less than 10%

In this paper, we propose the creation of a Challenge Dataset based on the *CheckList* paradigm for **Semantic Role Labeling**. This challenge dataset will be tested on two models. From now on, we refer to them as advanced and baseline.

2 Background

Semantic Role Labeling (SRL) is the task of classifying tokens or spans in a sentence according to the semantic function they perform or the *conceptual relation* they convey concerning the predicate (Palmer et al., 2011). The conceptual or *semantic* relationship is sometimes called *semantic role*. Depending on the *Grammar Theory*, the predicate can be everything that is not the subject or a particularly delimited span. As such, what constitutes an *argument* varies. In (Chomsky, 1993), Chomsky states that the relation between noun phrases and *semantic roles* is **inyective**. However, there are cases where this does not hold. In SRL, we are currently concerned with the core concept: *"...mapping from the syntactic analysis of the sentence to the underlying predicate-argument structures...* (Palmer et al., 2011).

The interpretation of which *semantic class* a predicate expression belongs to and what its *arguments* varies depending on the linguist framework. However, some terminology is shared among them. The term *theta-grid* names the set of *semantic roles* associated with semantic positions for a given predicate expression (Palmer et al., 2011). That is, the possible classes arguments can take. Different predicate expressions can share the same *theta-grids*, and as such, *semantic classes* can be constructed from these. FrameNet (Baker et al., 1998) and VerbNet (Schuler, 2005) and PropBank (Kingsbury and Palmer, 2002) make use of *theta-grids* to establish different *semantic classes* of varying granularities.

2.1 Capabilities

The capabilities of a system for identifying SRL are many and possibly unlimited. In (Ribeiro et al., 2020b), three basic capabilities are listed: a) Vocabulary, b) NER, and c) Negation. Vocabulary is vital in SRL, as unknown words lead to incorrect classification. NER might be helpful for correct identification of *ARG1*, which is usually "agents, causes, or experiencers" according to the PropBank annotation guidelines (Bonial et al., 2010). Negation is a critical capability, as one of the argument labels is *ARGM-NEG*, which relates to negations. However, they are not a complete set of capabilities. SRL intersects with the previously proposed capabilities in a very general way, as vocabulary can be applied to all NLP tasks. NER and Negation certainly play a role, but they are not central. SRL is more concerned with how the sentence is conceived syntactically and lexically, which will ultimately alter the semantic meaning. For instance, switching the places of two names in a sentence can completely change the meaning. Role labels can pertain to concepts such as *source* and *target*, which, when swapped, would change the label per token. The ability to detect these changes and select the new appropriate semantic role for the token is a core aspect of SRL. Generally, these alternations will carry a shift in the semantic flow of the sentence. Therefore, the capability tests proposed will be related to ensuring that the SRL system adequately adjusts to semantic changes via a lexical or syntactical alteration. Notably, we will be concerned with capabilities related to two subtasks: 1) *Identification* and 2) *classification*

2.2 Methodology

In this paper, we use PropBank and VerbNet *semantic roles*; however, other sources such as WordNet or FrameNet can aid in the development of capability tests. FrameNet helps generate instances of tests for different situations. The grouping of predicates by frame provides a way to check if a change in the predicate produces a change in the semantic role. A similar argument can be made for WordNet. The *CheckList* framework is helpful for automatically generating many test instances, but the information from VerbNet, WordNet, or even PropBank itself is needed to develop the test cases. VerbNet is mainly used to add information about the frames in each sentence used for testing. This annotation is done manually by carefully reviewing the descriptions in VerbNet and choosing the appropriate frame for the sentence. PropBank is used as the training data for the models to be evaluated, and the test subset is used to acquire more information about the accuracy of the models in a regular evaluation

environment.

3 Challenging SRL

Table **??** shows 10 tests for different capabilities. The tests can be divided into four broad categories: **Positional conditions**, **Noisy inputs**, **Edge cases**, and **General cases**.

3.1 Positional conditions

CA1-R, **CA0-D**, **CA1-D** test the identification capability on *ARG0* and *ARG1* under different positional conditions in regards to the predicate. One of the capabilities of an SRL system is to identify the predicate's arguments. This procedure should hold even for cases where the argument is far away from the predicate (> 3 words) or independent of whether it is located to the right or left of the predicate. **CA0-D** tests for distances bigger than three tokens to predicate on *ARG0* (*Agent*) while **CA1-D** tests the distances bigger than three tokens on *ARG1* (*Patient*). In (Xue and Palmer, 2004), features such as distance from predicate are used to train on SRL. As an example, sentences such as *"It can send a fighter squadron to strafe terrorist hideouts in the Bekaa Valley, but cannot shoot Abu Nidal"*, the predicate (in red) is separated from the argument *ARG0* (in blue) by a large number of tokens. We are treating a special set of sentences on **CA1-R**'s particular case. We propose a sentence that starts with *there* or *here* where the *patient* is located at the right of the predicate. These tests challenge the capability of correctly identifying the *ARG1* when the relative location regarding the predicate has changed.

3.2 Noisy inputs

CA1-N tests the classification of *ARG1* when noise is added at the end of the sentence. This noise is in the form of additional references to the subject while maintaining lexical, syntactical, and semantically coherence. This capability is concerned with identifying the correct *patient* in a situation where it could be unclear who the *patient* is. Similarly, as with **CADIR-P** capability, the idea is to introduce another possible contender for the label so that classification has to actually be informed and is not done simply by recognizing words related to patients.

3.3 Edge cases

CANEG-A tests the correct classification of negation adverbials as negation semantic roles (*ARGM-NEG*) instead of *ARGM-TMP*. According to PropBank documentation (Kingsbury and Palmer, 2002), given that an argument is both *ARGM-TMP* candidate and a negation, it should be classified as negation with higher priority. In this case, negative adverbials such as *never* and *rarely* test the model's ability to learn the hierarchy in semantic roles. The *ARGM-NEG* semantic will always have priority over the *ARG-TMP* semantic role.

CADIS-M tests discourse marker classification for *and* and *but*. *CADIS-M* is the argument assigned to discourse markers. The proposed test identifies the particular discourse markers of *and* and *but*.

3.4 General cases

CALOC-L tests the classification of *ARGM-LOC* when different location names are used. The goal of the test is to find out which is the extent of the identification of a location given different sentence contexts. Consequently, we seek to test if a location identified by a proper noun is not expected. The classification shifts to a different alternative, such as *ARGM-DIR*. This test could also be conceived as a robustness test against proper names.

CATMP-D tests the classification of varying ways to represent a temporal location. A date can be represented as a particular mention of a day, as a reference to the future like *'tomorrow'*, or even as a numerical date. This test evaluates the system against permutations of the temporal location for all months of the year and days of the week.

CATMP-F tests classification on a set of frequency adverbials. An SRL system should be able to accurately categorize frequency adverbials as *ARGM-TMP*, given that the frequency adverbials occur in different tenses. The **CATMP-F** test has three variants: one in the present tense, one in the past tense, and one in the future tense. It makes use of adverbials such as *tonight*, *last week* or *this afternoon*.

CADIR-P tests for variation in the prediction of *ARGM-DIR* in contexts where either the *source* or *goal* argument is provided and compares it to when the complementing (source or goal) is added at the end. For example, *The cat jumped from the roof*, paired with *to the university*. The first *ARGM-DIR* classification should not change by adding the other *ARGM-DIR*. Consequently, there are two tests for this capability. One for original sentence

containing a *goal* and one for original sentences containing a *source*.

3.5 Examples of tests

The bold code in each item refers to each test as per Table 1. The tests contain examples with the predicate color in red and the argument being tested for in blue.

- **CA0-R**
  1. There is the bus
  2. There are many cars
  3. Here come the warm jets
- **CA0-D**
  1. It can send a fighter squadron to strafe terrorist hideouts in the Bekaa Valley , but can't shoot Abu Nidal.
  2. The car can be incredibly fast and speed through different tracks without a problem.
  3. A student with good grades studies often.
- **CA0-D-2**
  1. Somebody was becoming less socially resilient
  2. Somebody could be less physically fast
- **CA0-N**
  1. John broke the window which was located underneath the lamp
  2. John shot the window which was located underneath the lamp
- **CALOC-L**
  1. I bought flowers in Miami
  2. My mother used to live in England
- **CANEG-A**
  1. I Never cook with an iron pan
  2. I said I would never
- **CADIS-M**
  1. But for now they look forward to the summer
  2. I looked and I could not find her
- **CATMP-D**
  1. My birthday is on Monday
  2. I visited my grandmother last Tuesday
- **CATMP-F**
  1. My birthday is tomorrow
  2. I visited my grandmother recently
- **CADIR-P**
  1. I want to go home
  2. My neck snapped back when I smelled that odor

4 Creating a Challenge Dataset

Creating a complete and meaningful challenge dataset is time-consuming. However, some subtasks can be automated. MFTs generally rely on a pattern that is slightly permuted. This characteristic makes them a good candidate for automation. Other types of tests required more care to be automated correctly. As previously mentioned, *CheckList* is a valuable tool for automating the process of generating challenge datasets. We used *CheckList* to ease the development of the proposed challenge dataset to different extents. Following is a brief recount of the procedure and the challenges faced when leveraging *CheckList* for SRL examples.

CA0-R, **CA0-D**, and **CA1-D** are good candidates for automation because they are MFTs, a general sentence structure, and a small list of predicates is enough to generate an extensive set of examples. However, in practice, predicate and argument identification becomes time-consuming if much variation is introduced. Therefore, we used simple templates that permitted slight variation while maintaining the structure to keep manual annotation minimal. The output of this procedure is a dataset with slight variation in terms of sentence composition. The changes introduced are mainly related to the predicate and subject in question. Even though *CheckList* provides a masked-model approach to generating examples, it proved to be more cumbersome than helpful in practice. In many cases, the predictions need to be semantically or syntactically correct. We reduced the number of predictions of the masked model to a minimum to mitigate the variation and errors. Additionally, we executed masked-model suggestions on different verbs and adverbials, which we filtered manually to later reuse in the *CheckList* template. The annotation of the position of the predicate and the argument was simple, as variation in examples did not affect it.

Code	Broad Capability	Specific Capability	Test Type
CA0-R	Classf. ARG1	**Role of patient** when argument is on the right	MFT
CA0-D	Classf. ARG0	**Role of agent** on distance > 3 to predicate	MFT
CA1-D	Classf. ARG1	**Role of patient** on distance > 3 to predicate	MFT
CA0-N	Classf. ARG1	**Role of patient** on added noise	DIR
CALOC-L	Classf. ARGM-LOC	**Role of location** for different locations	INV
CANEG-A	Classf. ARGM-NEG	**Role of hierarchy** of negative adverbials	MFT
CADIS-M	Classf. ARGM-DIS	**Role of discourse markers** *and* and *but*	MFT
CATMP-D	Classf. ARGM-TMP	**Role of temporal constituents** on different dates	INV
CATMP-F	Classf. ARGM-TMP	**Role of temporal constituents** on different adv. of frequency	INV
CADIR-P	Classf. ARGM-DIR	**Role of direction** on different places	INV

Table 1: Capability Tests

CALOC-L contains very few examples but with the possibility of many variations. Since the only change made was to use proper nouns in the sentence. Manually annotating sentences with the position of the predicate and argument, which were then perturbed on the mentioned names, proved sufficient.

CANEG-A is a set of simple worded sentences in the first or second person stating a particular action. Then, negative adverbials are applied cross-productively to all the sentences. Each of them shares the same length, argument, and predicate positions.

CATMP-D and **CATMP-F** also contain hand-curated examples manually annotated for the predicate and argument position. Given that the permutation is on a particular date, **CATMP-D** was simple to construct. **CATMP-F** uses a similar set of sentences constructed from common temporal adverbials that do not carry a negative connotation so as not to interfere with the *ARGM-NEG* specification. As mentioned, three test variants use temporal adverbials in different tenses.

CADIR-P contains two tests for evaluating if the classification remains invariant in two situations: 1) a sentence with a source as *ARGM-LOC* is concatenated with another which contains an argument that should be classified as *ARGM-LOC* but based on it being a goal, 2) the inverse case. In this case, sentences were manually annotated for predicate and argument location, and all combinations of base sentences and addendums were generated.

Additional information is needed to test the advanced model. Since the proposed model uses the base form of the predicate and frame number as input, each test that was not trivially generated required the manual annotation of the frame according to VerbNet. Consequently, for this particular task, *CheckList* presented itself as an obstacle in many different parts of the development process.

As mentioned previously, a few examples can be automatically created with *CheckList*, particularly in the case of SRL. Introducing variation introduces more complexity, which defeats the purpose. As such, many tests account for straightforward challenges. Also, it is worth noting that the converse happens for **CA0-R**, **CA0-D**, and *CA1-D*. Since they use masked-model predictions, some generated sentences have an odd feeling or are slightly semantically incorrect. These tests are complicated in this case as they do not provide a clean testing method. **CA0-N** and **CADIS-M** were not implemented due to their complexity and because a more precise methodology for test design would be required.

Developing platforms where more complex examples can be created would prove helpful to the community. During the development of these datasets, additional functionality was appended to the existing perturbation functions in *CheckList*'s library, which could have already been added based on simplicity. For that matter, it is essential to weigh the effort that requires the augmentation of a relatively simple system against developing a more catered generation script. While the latter might seem intuitively worse, in particular cases, the time

LR	Epochs	Batch Size	Weight Decay
$2e^{-5}$	2	32	0.01

Table 2: BERT Hyperparameters

it takes to generate the system from scratch to produce tailored examples is less.

5 Evaluating models

The models being evaluated are two BERT-based transformers. The baseline model is a simple instance of DistilBERT (Sanh et al., 2019) trained on sentences where the predicate is added to the input as in the question/answer pairing method. For example, a tokenized sentence would look like this: *"[CLS] His run for governor [SEP] run [SEP]"*. This method is similar to the one proposed in (Shi and Lin, 2019), where authors use a simple method to highlight the predicate. The other model seeks to provide BERT with additional predicate information. It adds the frame of the predicate along with its frame number. An example of the same sentence is: *"[CLS] His run for governor [SEP] run run.02 [SEP]"*. In the last case, the transformer now has information on the base form of the predicate and the particular instantiation of the frame. The purpose of including this is to have a marker that permits disambiguation where different predicate frames could be valid.

Table 2 shows the hyperparameters used for fine-tuning both models. We expect the models to perform relatively well on the simple tests related to permutations, while the more complex ones should give them some trouble. Additionally, given that the advanced model has information about the predicate frame, it should be able to disambiguate situations where the baseline model would fail, thus achieving a lower failure rate for the tests.

6 Results

Table 3 and Table 4 show the failure rate, capability, test number, and number of examples per test for the baseline and advanced models, respectively. The first three tests are very similar in both models. As speculated earlier before, tests **CA0-D** and **CA1-D** are either too complicated or most likely present structural deficiencies that trick the model. Both have a failure rate of 100%. **CA0-R** is the test with the lowest failure rate in both models.

The three tests related to capability **CATMP-F** offer very different results. Given that each test only represents a variation in the tense of the sentences, going from past to present ordinary with the number of the test. The past test (**T1**) has the highest error, followed by the future (**T3**), while the present tense has a 20% failure rate in the advanced model and 40% on the baseline. It is also the only test where the advanced model has a lower failure rate.

The **CADIR-P** capability tests also have a high failure rate in both models, having both 100% in both models on the first test and 88% and 100% in the second test for the baseline and advanced model, respectively.

The other tests oscillate between 30% failure rate to 80% in both models. Most of the tests have relatively high scores. However, note that the sample size is small, and examples within a particular test tend to be very similar.

Table 5 shows the *F1* scores of both models on the semantic roles evaluated with the challenge dataset. Each pair of rows corresponds to a given semantic role, and under *F1*, it shows the score for that particular label. The best scores are or *ARGM-NEG*, followed by *ARG0* and then *ARG1*. The lowest scores are on *ARGM-TMP*, with a deficient score for both models. The *F1* score in every semantic role tested in the challenge dataset is better in the baseline model than in the advanced model.

Capability	Test	FR %	Examples
CA0-R	T1	0	25
CA0-D	T1	100	25
CA1-D	T1	100	25
CALOC-L	T1	30	30
CANEG-A	T1	60	50
CATMP-F	T1	81.25	16
CATMP-F	T2	40	15
CATMP-F	T3	50	12
CATMP-D	T1	80	60
CADIR-P	T1	100	25
CADIR-P	T2	88	25

Table 3: Baseline Model

7 Discussion

Possible shortcomings and limitations were already hinted at in the previous sections on the construction and design of the challenge dataset. They mostly pertain to an accurate, varied sample of a capability. The scope of the capabilities has been lim-

Capability	Test	FR %	Examples
CA0-R	T1	4	25
CA0-D	T1	100	25
CA1-D	T1	100	25
CALOC-L	T1	56	30
CANEG-A	T1	62	50
CATMP-F	T1	100	16
CATMP-F	T2	20	15
CATMP-F	T3	50	12
CATMP-D	T1	83	60
CADIR-P	T1	100	25
CADIR-P	T2	100	25

Table 4: Advanced Model

Model	Argument	F1
Baseline	ARG0	0.82
Advanced	ARG0	0.8
Baseline	ARG1	0.81
Advanced	ARG1	0.75
Baseline	ARGM-NEG	0.92
Advanced	ARGM-NEG	0.87
Baseline	ARGM-LOC	0.6
Advanced	ARGM-LOC	0.51
Baseline	ARGM-TMP	0.0026
Advanced	ARGM-TMP	0.0017
Baseline	ARGM-DIR	0.39
Advanced	ARGM-DIR	0.21

Table 5: Scores on test set

ited as much as it remained coherent; however, to draw firm conclusions, more tests would be needed, as the results hint.

Table 3 and 4 show a significant difference between the scores of the **CATMP-F** capability. The variation in these tests is mainly related to the tense. This results in a difference of 60% of failure rate between the lowest scoring and the highest scoring tests. This behavior indicates that the variety in testing is significant; the failure rate can be affected by comparatively simple changes in the structure of the sentences, such as tense, more than with the particular predicates or nouns used. Nevertheless, many of the tests display high scores despite being relatively simple. This behavior indicates that neither model has developed a deeper and more accurate latent representation of the linguistic structures inherent in semantic roles, which the failure scores show. An excellent score is achieved only on straightforward examples such as **CA0-R**. As stated previously, some of the difficulty of the tests stems from their complexity or ambiguous semantic nature, such as **CA0-D** and **CA1-D**, for which neither model could make at least one correct prediction on the tested argument.

The worst performance of the advanced model could be either containing a number concatenated with the frame word, which is not correctly identified as an interpretation of the predicate in a different context, or perhaps being more of an obstacle. Other techniques to effectively convey more information about the predicate relation with the sentences should be tackled when developing SRL systems. Examining the *F1* score comparison of the models in Table 5 makes it more evident why there is such a clear difference between models. The baseline model initially performed better in the test dataset, suggesting its performance will improve on the challenge dataset. The case of **CATMP-F** under the test **T2** remains a mystery, having performed better in the worse model according to the test dataset.

As to *CheckList*, it proved to be a helpful tool when automatic basic programming behavior and during very restricted and limited cases. However, as is the case in SRL, when the end goal does not quite fit with *CheckList*, it, in turn, becomes cumbersome and staggers the progress of the examples generation.

8 Future Work

Inference on out-of-domain data affects all areas of machine learning (Rajagopal et al., 2019; Huang and Yates, 2010; Hartmann et al., 2017). In NLP, systems tend to drastically drop their performances when performing inference on corpora out of their training domain (Rajagopal et al., 2019). While datasets attempt to have a balanced set of data from different domains so that models can be trained and tested for good generalization, a solution currently needs to be found. In (Hartmann et al., 2017), a dataset is constructed to tackle the shortcomings in NLP systems. This includes that datasets are curated and normally do not include poorly written text or syntactical errors. In (Rajagopal et al., 2019) Authors tackle adapting SRL for low-resource specific domains in biological processes. Here, they propose a manual mapping from annotators to a considerably smaller set of labels that apply to the particular domain. They propose that it remains efficient since manual annotation has to be done only once per dataset. Additionally, different changes in architecture are proposed to increase the domain adaptability further. Other authors proposed a similar method for biomedical data, like in (Dahlmeier and Ng, 2010). However, these approaches require at least big domain-specific datasets and some form of manual annotation. However, different approaches, such as increasing the complexity of the model, are also present. Such as in (Huang and Yates, 2010) where Latent Variable Language Models are used to tackle domains where training data is small.

Many things could be improved when developing a system to target out-of-domain data predictions. The approaches mentioned above take a particular stance and tackle one problem. The use of different models along with data augmentation techniques is required to tackle such broad topics. Given that medical data and NLP tasks over medical data are rising, we will focus on that. First, a performance evaluation of our baseline model should be measured after being trained on our in-domain sample of PropBank. More robust measures than the proposed failure rate should be used to identify the actual size of the gap between the target domain and the current domain. In (Wang et al., 2020), authors examine the use of Maximum Mean Discrepancy (MMD) to measure the distance between two distributions as a metric to apply in domain adaptability. Given our robust measurement, it should be coupled with a challenging dataset. As we have seen through this work, apparently robust systems can fall prey to simple examples. Additionally, systems bridging the gap between out-of-domain and in-domain training data are paramount (Ben-David et al., 2010). Reducing this distance will result in systems that can learn better from small datasets and thus improve performance. Ultimately, the more available tools to challenge the trained systems will be beneficial as their actual performance will be precise.

9 Conclusion

Semantic Role Labeling is still an open problem in the field of NLP. Depending on which framework of semantic roles is chosen, the task can get increasingly more complicated. State-of-the-art models perform well on a relatively coarser set of roles like PropBank's. However, the test datasets of these models need to portray their performance accurately. Different authors have proposed techniques for further testing NLP systems to be able to portray their performance. One such approach is the use of *challenge datasets*. This dataset type is small compared to test datasets and is created to measure particular instances or fringe cases. They aim to establish theoretically motivated examples that only a system that correctly learned the task could identify. As such, these examples are related to particular linguistical phenomena in the area of NLP. Given the broad reach of SRL, we propose a challenging dataset to tackle ten different capabilities we consider paramount to an SRL system. We implement eight of those capabilities in the form of eleven tests and evaluate a baseline fine-tuned BERT model on SRL, as well as a more advanced version of the model fine-tuned on the same dataset. To do this, we use the *CheckList* library, which helps develop Minimum Functionality Tests and Invariance tests. Ey will target both functionalities that are required in the system and how the system fares against the perturbation or addition of semantic relations to the sentences.

Despite the apparent simplicity of the proposed tests, the failure rates are high for both the base model and the advanced version. In some cases, switching the tense of the sentence from present to past can drastically change the failure rate. Also,

adding additional possible arguments to the sentence with the same semantic role as an already present argument showed that failure rates increased. Finally, the advanced model had a more significant failure rate for all except one test where temporal adverbials were tested in sentences in the present tense.

References

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

Boris Beizer. 1995. *Black-box testing: techniques for functional testing of software and systems*. John Wiley & Sons, Inc.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning*, 79:151–175.

Claire Bonial, Olga Babko-Malaya, Jinho D Choi, Jena Hwang, and Martha Palmer. 2010. Propbank annotation guidelines. *Center for Computational Language and Education Research Institute of Cognitive Science University of Colorado at Boulder*.

Noam Chomsky. 1993. *Lectures on government and binding: The Pisa lectures*. 9. Walter de Gruyter.

Daniel Dahlmeier and Hwee Tou Ng. 2010. Domain adaptation for semantic role labeling in the biomedical domain. *Bioinformatics*, 26(8):1098–1104.

Arun Das and Paul Rad. 2020. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*.

Silvana Hartmann, Ilia Kuznetsov, Teresa Martin, and Iryna Gurevych. 2017. Out-of-domain FrameNet semantic role labeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 471–482, Valencia, Spain. Association for Computational Linguistics.

Fei Huang and Alexander Yates. 2010. Open-domain semantic role labeling by modeling word spans. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 968–978.

Paul R Kingsbury and Martha Palmer. 2002. From treebank to propbank. In *LREC*, pages 1989–1993.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2011. *Semantic role labeling*. Morgan & Claypool Publishers.

Dheeraj Rajagopal, Nidhi Vyas, Aditya Siddhant, Anirudha Rayasam, Niket Tandon, and Eduard Hovy. 2019. Domain adaptation of SRL systems for biological processes. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 80–87, Florence, Italy. Association for Computational Linguistics.

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

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

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020a. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020b. Beyond accuracy: Behavioral testing of nlp models with checklist. *arXiv preprint arXiv:2005.04118*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Karin Kipper Schuler. 2005. *VerbNet: A broad-coverage, comprehensive verb lexicon*. University of Pennsylvania.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Wei Wang, Haojie Li, Zhengming Ding, and Zhihui Wang. 2020. Rethink maximum mean discrepancy for domain adaptation. *arXiv preprint arXiv:2007.00689*.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *EMNLP*, pages 88–94.