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 a long-standing issue in different areas of machine learning (Das and Rad, 2020), and natural language processing is no exception (Belinkov and Glass, 2019). The difficulty arises in defining and materializing what a good *test* constitutes. A standard setup is to train a model and use a held-out part of the training information to test whether the model learned the underlying distribution or the underlying phenomenon. Other times, the testing data will come from a different distribution, for example, training on newspaper articles and testing on medical journals. Both settings can have their pitfalls since superficial or non-generalizable characteristics may be acquired or shortcuts learned (McCoy, 2019), generating a bias in the model (Rajpurkar et al., 2018). Additionally, phenomena such as *distributional shift* are not considered when models are evaluated for performance (Goel et al., 2021). The abuse of shortcuts or underlying casual correlations can also be present in the test data, thus making this metric unreliable since the outcome will be a heuristic that works but is not necessarily a representation of the phenomena.

1.1 Behavior testing and Challenge Datasets

An alternative to the previous techniques for model evaluation is behavior testing (Ribeiro et al., 2020a). It is a method that stems from software engineering and tests the model under particular conditions to ensure it is functioning correctly. It is analogous to *stress testing* the model to see where it breaks (Khurana et al., 2021). Behavior testing focuses on inputs and outputs while keeping the model as a black box (Beizer, 1995). However, due to the nature of the procedure, tests are highly specialized and domain-dependent. Creating such tests requires domain knowledge and time to handcraft examples. These manually-curated test sets are called *challenge datasets*. *Checklist* (Ribeiro et al., 2020b) aims to solve the issue of generating *challenge datasets*. It is a tool for behavioral testing on NLP models , which eases the development process by introducing ways to automate it.

1.2 Checklist

In *Checklist*, the authors establish the usefulness of this tool in three particular 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)

They name the performance of a model of a challenge dataset *failure rate*. This value is denoted by the percentage of tests that the model fails. There will be several tests for each task, and the organization for them is as follows. 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). Where each type of test serves to stress-test a particular functional category

- **MFTs** (Minimum Functionality Tests): evaluate a specific case where the label is knonwn
- **INV** (Invariance) tests: a characteristic is usually added, altered, or *perturbed* and compared to the base case under the assumption that the classification should not change.
- **DIR** (Directional) tests: are related to alterations, but a particular shift in the probability of the label is expected, either up or down.

There are different methods available to generate tests. Some examples are permuting an existing dataset, creating manually curated samples, or using *CheckList* perturbation templates. The first two methods are self-explanatory. However, templates need more explanation. Templates are created based on particular 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 {mask} often"*, the output token of the prediction model will replace the tag in blue. To better illustrate how *CheckList* works, we present an example provided by the authors on the *Sentiment* (sentiment analysis) capability (row) using a *MFT* (column) on a particular sentence (cell): *"The company is Australian"* has actual label of **neutral** sentiment and can be created with the template *"The company is {nationality}*.

In this work, we propose the creation of a *Challenge Data* for SRL. To assess the extent of what our models learn in terms of semantic and syntactical characteristics of language, we will devise a set of tests that aim to challenge the understanding of the model. We will develop a set of tests using a checklist to inquire about the actual knowledge of SRL by two BERT-based models, where one has more information about the particular frame, and the other has information about the predicate alone. To test it, we follow the behavioral testing approach and use *CheckList* to develop our challenge dataset.

2 Background

Semantic Role Labeling (SRL) is the task of classifying tokens or sets of tokens (spans) in a sentence according to the semantic function they perform or the *conceptual relation* they convey concerning a particular predicate (Palmer et al., 2011). This conceptual relationship is called a *semantic role*. Thus, SRL consists of finding a map between a syntactic component of the sentence and the underlying predicate-argument structure (Palmer et al., 2011). Simply put, it is related to finding relationships as: "who did what to whom," "when," and "where" (Shi and Lin, 2019). The interpretation of which *semantic class* a predicate expression belongs to and what its *arguments* varies depending on the linguist framework. However, they share some terminology. The term *theta-grid* names the set of *semantic roles* associated with semantic positions for a given predicate expression (Palmer et al., 2011) or, in other words, 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.

2.1 Annotatation Format

There are different frameworks to assign semantic roles differing in the granularity of a class and how they are constructed. Among them are FrameNet (Baker et al., 1998), VerbNet (Schuler, 2005) and PropBank (Kingsbury and Palmer, 2002). Here, we focus on *Universal PropBank 2.0* (Jindal et al., 2022), a variation of PropBank, which aims to establish English semantic role labels to other languages through a neural approach. It consists of annotated predicates, *theta grids*, and semantic roles for a broad set of sentences. Its generation performance was tested against a manually annotated test set, resulting in a robust new dataset for SRL. The following experiments, analysis, and results will use the conventions and frame available in *Universal Propbank 2.0*. Consequently, we will develop a *challenge dataset* considering the information available in *Universal Propbank 2.0*.

2.2 Capabilities

Many capabilities of an SRL system intersect with other NLP tasks. First, we will review the capabilities identified by(Ribeiro et al., 2020a) to inspect which are related to SRL. Then, we will devise the set of capabilities an SRL system should have. The authors in (Ribeiro et al., 2020a) list three basic capabilities: 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.

An SRL system should, first and foremost, be able to **identify** which of the tokens in the input are arguments. After signaling that one of the particular predicates in a sentence is the focus, the system

should be able to match the related tokens from the remaining set to arguments. Second, an SRL system should be able to classify the identified argument, considering the predicate and the sentence as a whole, which will require the actual frame to be known. We propose three general capabilities that all SRL systems should follow when working correctly: **Positional Condition Invariance**, **Edge cases Equivariance**, and **General Cases Invariance**. All of them involve an aspect of identification and classification. Additionally, our challenge dataset's tests will be designed to test this capability under different scenarios. An example is the semantic role of concepts such as *source* and *target*; if the nouns swap places, then the semantic role assigned should be swapped accordingly, given that the *source* became the *target*. Different subcapabilities within identification and classification exist, so our core capabilities are not meant to be exhaustive.

3 Methodology

In this work, we evaluate the performance of two fine-tuned BERT-based models on a challenge dataset trained on *Universal PropBank 2.0* semantic roles. The version of BERT used is *bert-base-uncased*. The two models are loosely based on NegBERT (Khandelwal and Sawant, 2019) where one is the *baseline* approach, and the other one is the *advanced* model.

- **Baseline model**: This approach follows closely the fine-tunning process in (Khandelwal and Sawant, 2019; Shi and Lin, 2019) by including the tokenized *predicate* separated by a *[SEP]* token after the tokenized sentence to highlight it. Additionally, *toke_type_ids* denote the presence of the *predicate*. We average the distributions to manage predictions across word pieces representing one token and then pick the token with the highest probability.

- **Advanced model**: Same setup as the *baseline model* with an additional mention of the frame of the predicate appended to the lemma to allow for sense disambiguation.

In the *baseline* model, a tokenized sentence would look like this: *"[CLS] His run for governor [SEP] run [SEP]"* and there is a *token_type_id* of 1 in the position of the predicate in the "question" part of the sentences are 0s elsewhere. The *advanced model* will have an additional token representing the lemma of the predicate in addition to the frame number according to *Universal Propbank 2.0* An example of the same sentence is: *"[CLS] His run for governor [SEP] run run.02 [SEP]"*. In the last case, the model now has information on the predicate's base form and the frame's particular instantiation. The purpose of including this is to have a marker that permits disambiguation where different predicate frames could be invalid. We will not evaluate the semantic roles of type *discontinous* or *references*; however, the frame instantiation could become more relevant in those cases. In the rest of this work, we will treat these roles as equal to their base form, such that *C-ARG1* (discontinuous ARG1) and *R-ARG1* (reference ARG1) will be equivalent to *ARG1*

LR	Epochs	Batch Size	Weight Decay
$2e^{-5}$	10	32	0.01

Table 1: Model hyperparameters

Table 1 shows the hyperparameters used for fine-tuning both models. We expect the models to perform relatively well on the simple permutation tests. 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.

3.1 Core capabilites

Table 2 summarizes the ten tests for different capabilities. We will detail each particular test for all four categories. They will be mentioned by specific capability, and the test size for ease of understanding following the convention in (Khurana et al., 2021). We explain each category and each test in terms of the capability it tests and argue why it is an SRL system capability. There will be examples in each capability test to show how these capabilities manifest. We denote the predicate using color red and the argument in color blue in addition to a subscript to indicate the *gold label*.

3.1.1 Positional Condition Invariance

PatientRight(n=25), *AgentRole3D(n=25)* and *PatientRole3D(n=25)* are meant to represent the capability of identifying and classifying the correct argument invariantly of the position. For*PatientRight(n=25)*, we are looking at instances

where the token to be classified takes the semantic role of *Patient(ARG1)*, and it lies to the right of the predicate. Expletive and depletive sentences show this kind of pattern. Thus, the model should correctly label these sentences with arguments that satisfy the predicate and the particular frame. *AgentRole3D(n=25)* and *PatientRole3D(n=25)* test the indentification and classification under far away arguments. In this case, we denote far away as having a token distance greater than three. In (Xue and Palmer, 2004), the authors use feature-based models to tackle SRL, and one of the features evaluated is the distance from the predicate. *AgentRole3D(n=25)* is concerned with the particular argument *Agent(ARG0)*. Consequently, *PatientRole3D(n=25)* is concerned with the particular argument *Patient(ARG1)*. Next, we show examples for each of the particular tests.

- **PatientRight(n=25)**
  1. There is the bus$_{[ARG1]}$.
  2. There are many cars$_{[ARG1]}$.
  3. Here come the warm .=jets$_{[ARG1]}$.
- **AgentRole3D(n=25)**
  1. Nobody $_{[ARG0]}$ that accidentally yet happily laughs in public is trustworthy.
  2. Anybody $_{[ARG0]}$ that intentionally and angrily drinks in the bus is unreliable.
  3. Anybody$_{[ARG0]}$ that accidentally yet happily cries is smart.
- **PatientRole3D(n=25)**
  1. The person$_{[ARG1]}$ next to that car is blurry.
  2. The child$_{[ARG1]}$ beside this road is happy.
  3. The mailman$_{[ARG1]}$ in front of that car is green.

3.1.2 Negation Equivariance

NegAdv(n=10) is related to equivariance under different negation adverbials. The model should correctly interpret negative adverbials as a negation. Also, note that negative adverbials related to frequency, such as *never*, should be categorized under the same semantic role label as other negative adverbials. Under the PropBank guidelines, negation (*ARGM-NEG*) takes precedence over frequency (*ARGM-TMP*). As examples:

- **NegAdv(n=10)**
  1. I never$_{\text{ARGM-NEG}}$ go to the gym.
  2. He never$_{\text{ARGM-NEG}}$ goes to the gym.
  3. He never$_{\text{ARGM-NEG}}$ watches TV.

3.1.3 General Cases Invariance

LocationVar(n=30) tests the classification using different location names. Specifically, the labeling of *ARGM-LOC*. Here, the capability of interest is the invariance under locations identified by a proper and regular noun. When there are changes to this token, it should not change to *ARGM-DIR*, for example.

- **LocationVar(n=30)**
  1. I bought flowers in Miami$_{\text{ARGM-LOC}}$
  2. My mother used to live in England$_{\text{ARGM-LOC}}$
  3. There is a fantastic flower shop in China$_{\text{ARGM-LOC}}$

TempAdv(n=50) tests the capability regarding temporal or time adverbial invariance. 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. The label should remain as *ARGM-TMP* under permutation. We follow with some examples:

- **TempAdv(n=60)**
  1. My birthday will take place on Monday$_{\text{ARGM-TMP}}$
  2. I visited my mother last May$_{\text{ARGM-TMP}}$
  3. I will have a party next Saturday$_{\text{ARGM-TMP}}$

FreqAdvPast(n=16), *FreqAdvPres(n=15)* and *FreqAdvFuture(n=12)* evaluate different frequency adverbials under changes in tense. The system should accurately recognize the frequency adverbial labeled as *ARGM-TMP* in all tenses. It makes use of adverbials such as *tonight*, *last week* or *this afternoon*. In each test, a particular tense will permute to another frequency adverbial under the same tense; for example, in *FreqAdvFuture(n=12)*, we can have a change from *next week* to *next month*.

- **FreqAdvPast(n=16)**
  1. I visited my grandmother recently$_{\text{ARGM-TMP}}$

2. I ate a cake yesterday$_{\text{ARGM-TMP}}$
3. He slept last night$_{\text{ARGM-TMP}}$

- **FreqAdvPres(n=15)**

1. Her car is arriving today$_{\text{ARGM-TMP}}$
2. The party is starting tonight $_{\text{ARGM-TMP}}$
3. The movie premiers today$_{\text{ARGM-TMP}}$

- **FreqAdvFuture(n=12)**

1. I will have a party tonight$_{\text{ARGM-TMP}}$
2. I will go to the beach next week$_{\text{ARGM-TMP}}$
3. He will go to the park tomorrow$_{\text{ARGM-TMP}}$

VarDirTarget(n=10) and *VarGoalSource(n=10)* evaluate the capability related to correct labeling under the introduction of a source to the sentence. In the first, we expect that under the addition of a target, the token label (*ARGM-DIR*) remains invariant, and in the second, we expect that under the addition of a source, the token label (*ARGM-GOL*) remains the same. Concretely, with this test, we are trying to induce the model to label incorrectly with *ARGM-GOL* in the first case and with *ARGM-DIR* in the second case.

- **VarDirTarget(n=25)**

1. The cat jumped forward$_{\text{ARGM-TMP}}$ from the roof to the pool
2. I biked from my house$_{\text{ARGM-TMP}}$ to the university
3. The dog ran along the road$_{\text{ARGM-TMP}}$ to the gym

- **VarGoalSource(n=25)**

1. Workers dumped the waste from my house into a huge bin $_{\text{ARGM-GOL}}$
2. My brother threw a paper into the red basket $_{\text{ARGM-GOL}}$
3. The laundry should be put into the garbage disposal $_{\text{ARGM-GOL}}$

4 Challenge Dataset

The construction of the *challenge dataset* involved manual and automated components. The manual part involved finding examples of the particular capabilities of the tested and constructing base sentences so we could easily manipulate them. The automated part involved using *CheckList* to generate different permutations of sentences. We extracted the frame information from PropBank during this process to determine which predicates have which arguments. Some of them will not have *ARG0*, for example; thus, care needs to be had when picking the particular predicates. Additionally, sentences were annotated manually with the respective frame for the predicates since the advanced model will need this information. We focused on two types of tests: *MFTs* and *INVs*. The former is concerned with testing a specific functionality for which we know the answer. The latter introduces some variability in the input that should not change the output. At the same time, we labeled some tests as *INV* and others as *MFT*. This procedure is a pure categorical distinction since they will materialize in one input sentence and the expected output for a particular argument. However, they do help conceptualize the goal of the test. Table 2 summarizes the capability tests, the argument they target, and the type of test. Next, we go over each test individually.

PatientRight(n=25), *AgentRole3D(n=25)* and *PatientRole3D(n=25)* were good candidates for automation because the structure the sentences need to follow is not significantly constrained. We can plug in a small list of predicates with a defined structure and many pronouns, adjectives, and nouns to create various sentences. In the case of *PatientRight(n=25)*, sentences of the form *"There ..."* or *"Here ..."* (expletives) are natural candidates for positioning the patient on the right. To add distances from the predicate to the arguments, we introduced tighter templates, as more variation makes the tagging steps more manual or even heuristic, which defeats the purpose of automating the procedure. In these cases, we stick to one or two predicates and alter the other components of the sentences. Also, 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 were semantically or syntactically incorrect. However, we executed masked-model suggestions on different verbs and adverbials, which we filtered manually to later reuse in the template.s The annotation of the position of the predicate and the argument was simple, as variation in examples did not affect it.

Name	Target argument	Test Type
PatientRight(n=25)	ARG1	MFT
PatientRole3D(n=25)	ARG1	MFT
AgentRole3D(n=25)	ARG0	MFT
LocationVar(n=30)	ARGM-LOC	INV
NegAdv(n=10)	ARGM-NEG	MFT
TempAdv(n=60)	ARGM-TMP	INV
FreqAdvPast(n=16)	ARGM-TMP	INV
FreqAdvPresent(n=15)	ARGM-TMP	INV
FreqAdvFuture(n=12)	ARGM-TMP	INV
VarDirTarget(n=25)	ARGM-DIR	INV
VarGoalSource(n=25)	ARGM-GOL	INV

Table 2: Capability Tests

LocationVar(n=30) 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 and then perturbing on the mentioned names proved sufficient. Additionally, we filtered out locations' names with more than one word for convenience. Since BERT will convert everything to lowercase during tokenization, a capitalized letter for a proper noun should not affect the prediction but rather the actual token.

NegAdv(n=10) is a set of simple worded sentences stating a particular action in the first or second person. Then, negative adverbials are applied to the sentences.

TempAdv(n=60) contains three hand-curated examples manually annotated in which we will introduce variations of the days of the week and the months. With this setup, we were able to construct a large number of tests.

FreqAdvPast(n=16), *FreqAdvPresent(n=15)*, *FreqAdvFuture(n=12)* were constructed with a list of frequency adverbials for the three tenses. For each test, we will permute the frequency adverbial at the end with another frequency adverbial corresponding to the tense we are looking at. We were limited to a small number per tense because of the difficulty of constructing these types of sentences.

VarDirTarget(n=25) and *VarGoalSource(n=25)* were the hardest to construct. First, we wanted to make sure that *ARGM-DIR* and *ARGM-GOL* are present in the base sentences without the permutation. Then, we want to introduce a permutation that will not change that particular argument but add additional information that might confuse the model. However, something we also want to look at with this test is a functionality that we have not enforced. Namely, that numbered arguments take precedence over modifier arguments. We would like to see if the model will clearly label temporal arguments as numbered arguments in some cases.

As mentioned, we created a few examples with *CheckList*. Introducing variation introduces more complexity, defeating the purpose since we must manually tag arguments or develop complex heuristics. As such, many tests account for straightforward challenges. For our particular use case, we used the library to simplify typing and coming up with ideas. On the side of the integration, the usage was relatively simple. Since we had to perform postprocessing, exporting the test suite was more straightforward than running the tests directly via *CheckList*. Ultimately, we only had a tabular data file with the information we still had to process and pass to the model for inference.

5 Results

Table 3 shows the capabilities and respective failure rate per model. The numbers highlighted in blue show the best score out of the two models, and the numbers highlighted in red show when the score of the advanced model was lower. Consequently, if there is no red score next to a blue score, then the advanced model improves. In the other case, performance decreased.

First, in *PatientRight(n=25)*, we had an increase in performance related to one particular sample that was classified correctly. We introduced one manually annotated test to test if the model had learned expletives and the predicate *lives*. In the sentence *"There lived a person"*, according to *PropBank*, the word *person* is not a patient (*ARG1*) but an agent *ARG1*. The advanced model containing the particular *live.01* frame could make this correct classification. However, the base model failed. In *PatientRole3D(n=25)*, we have the only case where the advanced model did worse than the baseline model. It incorrectly labeled the argument as *O* or no argument. In *AgentRole3D(n=25)*, the performance was equal to zero failure.

NegAdv (n=10) was ideally classified with a failure rate of zero with both models, something to be expected given the simplicity of the examples.

LocationVar (n=30) also shows improvement under the advanced model. The baseline model fails when the country labeled as *ARGM-LOC* is *Uzbekistan*, and the advanced model succeeds.

TempAdv (n=60) In this case, the baseline models seem to perform much worse. The errors seem to be on different days of the week or month without an identifiable pattern. It is worth noting that with the included frame information, the advanced model has a failure rate of zero.

In *FreqAdvPast (n=16)* and *FreqAdvPresent (n=15)*, both models performed well even in arguments where the frequency adverbial was composed of two words so the headword had to be tagged. However, the baseline model struggled in *FreqAdvFuture (n=12)*. It is its worst score with a 50% failure rate.

VarDirTarget(n=25) and *VarGoalSource(n=25)* are particularly difficult tests because of the ambiguity. In some cases, the label could also be *ARG2*, where the models fail. The baseline model made two mistakes in the former, while the advanced model has a zero failure rate. In the latter, both models had a 100% failure rate. We devised examples based on the PropBank handbook and only introduced some variations. All examples were tagged with *ARG2* in both models. Consequently, it seems like this classification was challenging.

6 Discussion

The results of the challenge dataset reveal essential insights into the strengths and limitations of current SRL systems, mainly when tested under conditions designed to challenge their capabilities. The performance differences between the baseline and advanced BERT-based models show instances where adding linguistic information proves beneficial.

The failure rates observed in the baseline model across various tests indicate that, despite the model's performance in SRL tasks, it struggles with more complex linguistic phenomena. The advanced model generally outperforms the baseline model, which incorporates additional frame information about the predicate to disambiguate meaning. However, the improved performance of the advanced model is limited. In fact, in one instance, it performs worse than the baseline model, suggesting that the additional frame information might sometimes introduce ambiguity or noise rather than aiding disambiguation.

One of the key findings is the consistent improvement of the advanced model on tests where the particular frame of the predicate is needed to infer the arguments such as *PatientRight(n=25)* and *LocationVar(n=30)*, where its performance beat the baseline model in challenging examples. These tests involve tasks where the model has to generalize well across different locations and handle positional argument changes. These tasks benefit from the additional frame information available to the advanced model.

The poor performance of the baseline model in *TempAdv(n=60)* shows issues in recognizing variations of nouns, suggesting that the model is learning shortcuts or heuristics for prediction. The difficulties in handling these cases reflect an ongoing problem in SRL systems: a tendency to rely on surface features, such as word order, rather than fully understanding the underlying semantic rela-

Capability (n=x)	Model Name	Failure Rate
PatientRight (n=25)	Baseline	0.040000
	Advanced	0.000000
PatientRole3D (n=25)	Baseline	0.000000
	Advanced	0.040000
AgentRole3D (n=25)	Baseline	0.000000
	Advanced	0.000000
LocationVar (n=30)	Baseline	0.033333
	Advanced	0.000000
NegAdv (n=10)	Baseline	0.000000
	Advanced	0.000000
TempAdv (n=60)	Baseline	0.333333
	Advanced	0.000000
FreqAdvPast (n=16)	Baseline	0.000000
	Advanced	0.000000
FreqAdvPresent (n=15)	Baseline	0.000000
	Advanced	0.000000
FreqAdvFuture (n=12)	Baseline	0.500000
	Advanced	0.000000
VarDirTarget (n=25)	Baseline	0.080000
	Advanced	0.000000
VarGoalSource (n=25)	Baseline	1.000000
	Advanced	1.000000

Table 3: Comparison of Model Failure Rates (Best Score Highlighted in Blue)

tionships between the tokens.

One of the most surprising findings was the high failure rate for the baseline model in *FreqAdvFuture(n=12)*, even though it dealt with relatively straightforward temporal adverbials cases. The fact that altering tenses led to such a steep decline in performance suggests that both models might struggle to recognize temporal cues and effectively generalize across different time frames. This behavior raises questions about the models' ability to deal with temporality and their reliance on dataset-specific features learned during training, which may not transfer well to novel or out-of-distribution examples.

Additionally, the 100% failure rate in the *VarGoalSource(n=25)* test for both models highlights a critical limitation. The models are not equipped to handle more complex cases involving goal-related arguments. This failure could be related to inherent limitations of the BERT architecture or, more broadly, the transformer architecture since adding frame information was unsuccessful.

The results highlight the importance of challenge datasets. We might be fooled into thinking that we have a good model of the world when measuring the performance on test sets. However, as we have shown, this is in many ways different. Also, these tests do not aim to be comprehensive in all the capabilities in SRL but a sample of core capabilities that should be a minimum. As such, they are limited in complexity and reach. Moreover, adding *discontinuous* or *reference* arguments as labels or investigating less prominent labels might help shed light on deeper issues.

7 Future Work: Domain Adaptation

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), authors construct a to tackle the shortcomings in NLP systems. This process includes curated datasets that usually

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 suggest it remains efficient since manual annotation happens only once per dataset. Additionally, introducing different changes in architecture can further increase the domain adaptability. 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 techniques, 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. However, we approach this issue from the perspective of *challenge datasets*. We aim to reason about the procedure for devising a *challenge dataset* that provides information on the out-of-domain generalizability of an SRL model to the target domain of **history textbooks**. Naturally, we need to consider characteristics shared between the source and target domains. For example, we uncovered that models have limitations with less common country names such as *Uzbekistan*. Consequently, if our target domain is history books containing uncommon city or region names, we would like to test the model understanding of locations by employing uncommon location names in the *challenge dataset*. In this particular case, it will be essential that the model identifies them. More broadly speaking, we would like to test the ability of the model to identify proper nouns in all circumstances since names of essential figures, regions, titles, or specific things will be prevalent. Additionally, as we have seen in our investigation, disambiguation plays a significant role. The *challenge dataset* should aim to test ambiguous predicates by relying on differences in arguments of predicate frames. Since the models fail when one of the frame's meanings is prevalent, cases, where less frequent predicate frames are used, could induce a lot of inaccuracy in our predictions. As such, testing not only particular linguistic phenomena, such as what we have done in this work, but also exploring broader differences in meaning is essential. Finally, since we are looking at large measurements of unlabeled data, coupling the failure rate with a more robust statistic-based measurement could prove helpful. 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. Training a model with approaches that minimize distances to target distributions has proven useful, given that we do not fully know the distribution but only sample from it.

8 Conclusion

This study sheds light on challenge datasets' usefulness in probing language models for specific linguistic capabilities. While transformer-based models like BERT have made significant advancements in capturing syntactic and semantic relationships, the results demonstrate that these models still face substantial challenges when the phenomena are more complex. The advanced model, which incorporates predicate frame information, generally outperforms the baseline model, especially in tasks requiring disambiguation or positional invariance. However, even the advanced model struggles with more intricate scenarios, such as handling argument swapping, temporal adverbial variations, and directional arguments.

The high failure rates observed in tests like *VarGoalSource(n=25)* and *FreqAdvFuture(n=12)* highlight the limitations of these models in understanding complex semantic structures. These findings suggest that current SRL systems perform well on standard datasets and rely heavily on surface-level heuristics or shortcuts that fail to generalize to more complex or less common linguistic patterns. Moreover, the unexpected performance drops in seemingly simple tasks, such as *TempAdv(n=60)*, point to the need for models that can better grasp temporality and other nuanced aspects of language.

Our findings emphasize the value of behavioral testing using challenge datasets, as they expose significant weaknesses that might need to be evident in traditional evaluations. These datasets are critical for identifying gaps in SRL systems' semantic comprehension and revealing their dependence on the specific features of the training

data. As such, developing SRL models requires more than improving accuracy on typical datasets. They require more robust methods to test if models can generalize across diverse, real-world language uses, including edge cases and out-of-domain contexts.

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.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

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*.

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*.

Karan Goel, Nazneen Rajani, Jesse Vig, Samson Tan, Jason Wu, Stephan Zheng, Caiming Xiong, Mohit Bansal, and Christopher Ré. 2021. Robustness gym: Unifying the nlp evaluation landscape. *arXiv preprint arXiv:2101.04840*.

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.

Ishan Jindal, Alexandre Rademaker, Michał Ulewicz, Ha Linh, Huyen Nguyen, Khoi-Nguyen Tran, Huaiyu Zhu, and Yunyao Li. 2022. Universal proposition bank 2.0. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1700–1711.

Aditya Khandelwal and Suraj Sawant. 2019. Negbert: a transfer learning approach for negation detection and scope resolution. *arXiv preprint arXiv:1911.04211*.

Urja Khurana, Eric Nalisnick, and Antske Fokkens. 2021. How emotionally stable is albert? testing robustness with stochastic weight averaging on a sentiment analysis task. *arXiv preprint arXiv:2111.09612*.

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

RT McCoy. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

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. *arXiv preprint arXiv:2005.04118*.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020b. 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.

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.