Checklisting SRL

Gabriel Hoogerwerf
Vrije Universiteit van Amsterdam
g.hoogerwerf@student.vu.nl

Abstract

Standard automatic evaluations of NLP systems on a test split of data offer the advantage of comfortably comparing the performance between different approaches but little they reveal about the real world performance of these models. Studying in depth the behavior of a system given unseen or challenging input is in fact considered good practice and insightful for future improvements. For this reason, (Ribeiro et al., 2020) propose a framework called *Checklist* to help users and developers to plan and design a challenge data set that allow to singularly test the different capabilities required to resolve a certain task. In this work, following the *Checklist* approach, the main capabilities for Semantic Role Labeling are discussed and a set of test data is proposed based on them. The behaviour of two state of the art models is then evaluated on the set and their performance discussed. Results reveal that these models still struggle with some of their core capabilities when tested on these targeted test sentences.

1 Introduction

Researches in Natural language processing have seen great improvements in recent years, mainly thanks to the appearance of Neural models, unleashing systems able to process, understand and generate natural language with performances often comparable to humans. No matter the method adopted to tackle a specific problem or task, the evaluation of the system is crucial to prove its effectiveness and to be able to compare it with others. In the field of Artificial Intelligence (and NLP is no exception), it is common to provide evaluations based on numerical metrics calculated against a held-out test data set. Too often though, this test set is not exhaustive or too similar to the training data to be able to tell us about more in-detail behaviour of the model against some of the exceptions and oddities that natural language contains in real world. A good alternative (or supplementary) evaluation approach are challenge data sets, which are purposely created to challenge a method by presenting exhaustive exceptions or targeting specific phenomena. This is can be called stress testing and it is very used in any kind of engineering process when testing a ”complete product”. Such test suits have actually been around NLP researches for a long time (Lehmann et al., 1996), but they are still not the standard practice in the experimental pipeline. In this work I propose a challenge data set for Semantic Role Labeling, a core task in the field of Natural Language Understanding. Various *capabilities* that any SRL model should have are discussed and targeting example cases are created to study such behavior in details. This approach is based on the work of (Ribeiro et al., 2020), in which the authors propose a framework called *Checklist* to generate synthetic data, and showcase it for different NLP tasks. Two different SOTA performance models are tested against the generated data and the results are compared.

The rest of this document is structured as follows: First an introduction to *Checklist* is given and how it can be used to test the behavioir of NLP models. In section 3 I will provide some background knowledge about Semantic Role Labeling (SRL). I will then discus the capabilities that the proposed tests aim at verify as well as tests organization. In section 5 the Data set generation process and the various techniques are explained along with the implementation details of the tests if necessary. Sections 6 and 7 contain respectively the Experiment conducted and the Discussion. Section 8 Future work and finally the conclusions.

2 Checklist

In this work, the authors present a methodology for behavioral testing of NLP models along with a software tool to automatically generate a large number of tests following a template. By showcasing it on three NLP tasks performed by up to 5 different models, Ribeiro et al. demosnstrate how to exhaustively test any NLP model by following behavioral testing (or black box testing) schemes inspired from software engineering. Here thought, the individual components of the more complex system are reflected by the different capability that a NLP model needs to have in order to successfully carry on a task. Exhaustive implementation-independent testing is a fundamental step of every development process and there are often entire teams whose work revolves around designing and creating such test suites. This work's main contribution can be thought as not only a conventional methodology for implementing behavioral tests and a very accessible framework to put it in practice, but also as an encouragement to both users and developers to turn more and more focus on alternative testing rather than the standard ML paradigm, simulating as much as possible real world scenario. The need for a better and in depth evaluation of a models comes from acknowledging that in fact, training data differ a lot from real world use cases (and this is often the case for test as comes from the same split). When it comes to neural models in fact, which entirely rely on data to learn how to elaborate different inputs and turn them into an output, it is of high importance that such data represents as much as possible the whole domain in which the model will be deployed. This is unfortunately not easy as training data often contain biases, especially if coming from a single or a small amount of sources and generalization remains one of the big challenges of machine learning. In practice, Checklisting a model consists of identifying the core capabilities for a task and designing tests from three different types: minimum functionality test (MFT), Invariance (INV) and Directional Expectation (DIR). In a typical Checklist matrix, rows represents the capabilities of interests while the columns the different test. It is not strictly necessary fill entirely the table. Each cell should be filled with the failure rate namely the percentage of failed tests over the total number. In the paper, three NLP tasks are picked to showcase Checklist. For each of them, the core capabilities are identified and a large scheme of test is proposed. The tasks are: Sentiment Analysis (SA), Quora Question Pair (QQP) and Machine Comprehension. Sentiment Analysis (SA) consists of labeling a document as Positive, Negative or Neutral based on the dominant sentiment in the text. This is useful for automatically labeling user's reviews of a product for example. For this task, an example of capability to be tested is Vocabulary+POS, by checking whether a model is able of identifying and value adjectives that influence the overall sentence/document sentiment. Machine Comprehension (MC), which is often formulated as a Question Answering problem, consists of giving a piece of text to a language model and successively querying it about the information provided. Coreference is among the core capabilities applied during the understanding process tested by the authors as it is necessary, in case of pronouns attachments, to resolute who/what the additional information are referring. Finally, QQP, also known as duplicate question identification, aims at recognizing when two questions are actually asking the same thing. This is important for general question answer skills as various different questions could be rephrased in different way but still require the same answers. To better portrait the necessary reasoning behind defining the capabilities and testing, we will discuss the test set proposed by the author of the papers for QQP. This task is far from being trivial as it requires a lot of high level understanding and processing skills. In the work, 9 capabilities are proposed and tested. The most significant/interesting ones are hereby discussed. *Vocaulary* is crucial and usually the appearance of a new term in the questions introduce a relevant change. This is tested with an MFT. *Taxonomy*, revolves around word categories and their relations. For this task, it is crucial that a model is able to handle synonyms or antonyms, i.e. If a similar word/expression is used, the question is likely not to change its meaning. *Temporal* understating is a very high level capability and is again crucial for the task at hand: If the question is about or revolves on the temporal information of an events, the model should posses such awareness. This is tested by asking the model to compare two questions containing different temporal expression (before and after) or different tenses of the same verb. *Negations* tend to invert the meaning of a sentence and the detection and understating of those is almost mandatory for a similarity detection tasks: A simple MFT comparing a question and its negated version reveals if the model is able to tell them apart. *Coreference resolution* is fundamental when understating language as relations in the text might involve the same subject or object but for repetitions sake these are replaced with pronouns. Associating the pronoun to the correct entity is thus crucial for understand what is the question referring to. The authors utilizes similar sentences that only different in the pronoun (which is enough to create a different question) and let the model predict their expected dissimilarity.

All those tests showcased in the paper are created with the python library publicly available and implmented by the authors. It allows to fill the gaps (masks) le ft in a manually generated templates with a customizable list of options and creates a large amount of sentences based on that templates up to all the possible combinations (the Cartesian product) of all the options for every gap. Moreover, the users can take advantage of a lexicon class which provides large list of names, cities, adjectives etc. It is further possible to use a masked language model (RoBERTa (Liu et al., 2019)) predictions to fill in a mask in the template with a a coherent choice (these models are purposley pretrained on masked language modeling task).

This approach as a whole is evaluated in a User-study conducted on users with some NLP knowledge, who were asked to test BERT with and without the help of *Checklist*. Results show how with the help of the framework, users were able to create twice as many tests and find almost three times more bugs than the users without it.

3 Background

Semantic Role Labeling is the process of assigning to every element of a sentence its semantic role within the period. It works thus at a sentence level and provides insights about the predicate and its argument. This task was first introduced in (Gildea and Jurafsky, 2000) and since then it has been widely researched because its importance and difficulty. Given the complexity and variability of sentences in fact, automatic SRL is far from being solved and still represents a challenge for every system that involves natural language Understanding. Figure 1 shows an example of a sentence where the semantic function of every span is found, in this case by answering the questions ”Who, What, to Whom and where?”. Although there is no universal convention shared among linguists on how to label different semantic components of a sentence, it is often good practice to identify the predicate (the verb) first and then identify its argument and their roles. This is because for every verb, there is a recurrent set of roles. For this reason 4 sub tasks usually constitutes SRL, namely predicate identification, predicate classification, arguments identification and arguments classification. Many corpora and lexical resources have been used in the literature as reference for labeling the tokens of the sentence, among these is worth mentioning Frament (Baker et al., 1998), Verbnet (Palmer et al., 2017) and PropBank (Kingsbury and Palmer, 2002). The latter is used as reference in this work and its notation for labeling roles is adopted. PropBank is an annotated corpora based on Penn Treebank (Marcus et al., 1993) and is based on the concept of *rolesets*. Rolsets are similar to the concept of semantic frames and they capture an instance of an action taking place described in the text. This action, as we will see, can not only be introduced by verbs but also by other syntactic components. To give an example, the main arguments identified in the PropBank annotation guidelines (Babko-Malaya, 2005) are shown in the table below.

ARG0	Agent
ARG1	Patient
ARG2	Instrument, Benefactive, Attribute
ARG3	Starting point, Benefactive, Attribute
ARG4	Ending point
ARGM	Modifier

Table 1: Core arguments identified in PropBank.

Lastly, it is worth mentioning that two main approaches are used regarding the label annotation and output of SRL systems: Span based and Dependency Based. These two methods have both their own perks and downs as widely explained in (Li et al., 2019), but it is important to be aware of the notation in use to be able to exploit the output of the system at hand. In this work, the span based notation is adopted.

4 Challenging SRL

Being SRL quite a complex and comprehensive task, many are the capabilities involved throughout the whole process, at various different levels of processing. It can be said though that it does not revolve around high-end natural language compre-

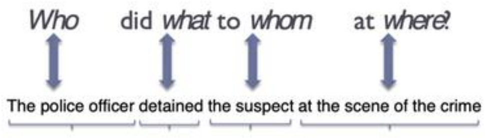
Who
did what to whom
at where?
The police officer detained the suspect at the scene of the crime

Figure 1: Example Sentence with roles

hension such as Logic reasoning, Temporal cognition of Fairness. Moreover, given the sentence centered nature of the task, capabilities such as sentences identifications, or relations between sentences are also not needed to carry out the task. Among the most common capabilities for the various example tasks proposed in the original Checklist paper, some like *Vocab+POS* (important words and their Part of Speech) and *NER* (properly identifying Named entities in the sentence) can be considered important for our task as well. It is further worth in our case investigating *Taxonomy* as when labeling argument, different words plays different semantic roles often because of their ontological meaning and a functioning SRL system should be able to tell them apart. If a token is a name of a person for example, or a nationality, it is not going to be carrying temporal information about the action taking place. This prediction should not (only) rely on order, surrounding words or syntax but on some kind of taxonomic knowledge as well. *Robustness*, tested in (Ribeiro et al., 2020) as resistance to typos might not be a core capability of SRL specifically but it will be investigated since being capable of dealing with any kind of different language perturbations is always useful for any NLP system. On this note, we will also test two capabilities based on such robustness assumptions namely *Grammar* and *Paraphrasing*: As SRL highly revolves around verbs and being verbs a very mutable and arguably the most complex particle to deal with, it is important that an SRL system recognize the predicate no matter the nature of the morphological inflection applied to the base form, thus some kind of grammatical knowledge of the various verbs and their behavior is required. When it comes to *Paraphrasing* on the other hand, no matter the words order, format, or a stylistic transformation of a sentence: a role set and the different arguments should remain the same. To test this, it would be appropriate to apply different transformation such as Direct vs Indirect speech or rephrasing the sentence as a question and check whether the arguments and the frame remain the same. In addition to those proposed in *Checklist*, I further introduce *Ambiguity* as a core capability of SRL. Language is by nature ambiguous and this can greatly affect performance in solving SRL. This capability can be tested at various different levels: a single token can represent a source of ambiguity in case of multiple meanings (*Polysemy* as well an entire span of tokens (*PP-attachment ambiguity*). I will be further investigating *Rarity*: This consists of testing the SRL with tokens that have been barely or never seen before such as slangs, old English or new words. It could be argued that RARITY as just described is just a sub Capability of *Vocabulary*. This point of view is not necessarily wrong, but I believe it is worth introducing a new set of tests on its own. Dealing with Rarities is the capability of the model to improvise and still perform well on data that were (almost) never seen and as language is constantly evolving and far from being repetitive and predictable, it is good to consider this ability for any kind of natural language processing technology. Finally, I argue that *Span Detection* can also considered an important capability of SRL. As mentioned above, some systems might or might not produce a span based argument identification/classification, but it could be argued that in order to classify every token, some kind of sentence parsing and grouping is necessary. Moreover, the models considered for the experiment do provide the full span with BIO tags annotation thus this capability can be easily tested.

As mentioned in task description, SRL can be divided in sub tasks, this division consists is predicate identification, predicate disambiguation, arguments identification and arguments classification. As our task is to create a set of tests to in detail analyze performance and behavior of an SRL systems, it is worth proposing, along with general capabilities, tests for this specific four sub tasks, in order get a closer look at how SRL behave in every of the small tasks to get an insight. It is worth noticing how these tasks are dependent, in the order they were mentioned, to one anther. In this work we therefore propose two different test tests.

4.1 Challenging Predicate Identification

To test the sub task of predicate identification, a test set revolving around the ability of correctly identifying the predicate in a sentence. This can be seen as a token binary classification task. As in this work we focus on SRL based on the PropBank annotation scheme, the predicate can be referred as a roleset and such roleset can be introduced by different words with different POS tags. Despite this, it is fair to say that in the majority of case the word we are looking for is a verb. Verbs are often quite mutable entities and English is not an exception. For this reason we propose two tests for testing *Grammar*: Contraction and Inflections. For contraction an Invariance test is proposed where two sentences differ only in the contracted particle of which the index is known. For Inflections I check with an MFT test if the system is able to recognize irregular inflected verbs.

For *Robustness*, various verbs are perturbed with typos to see whether they can still be detected by the model.

With respect to *Ambiguity*, it is the case in many languages that morphemes which corresponds to verbs also appear as nouns. The meaning might or might not be related. Examples of this are the words *"bank"* or "*book*". This phenomena is called polisemy, and can be considered as an ambiguity for the system. For this reason, a DIR test is introduced where two sentences containing the same word (but used with two different meaning) are expected to have different predictions assigned, specifically, in one sentence the token of interest must be identified as a verb whereas in the other must not. Another test to be proposed is the ability of distinguishing gerunds when used in nouns or adjective positions. It is sometimes the case in English that gerunds (supposedly a verb inflection) play the part of nouns for example, (i.e. they often refer to the action/activity itself). An example from the PropBank database: *"The **writing** of the section on aboriginal culture"*, where *writing* refers to the roleset *write.01*.

Finally, for verb identification I introduce *Rarity* tests, by checking whether the system is able to detect slang verbs, that presents a rare form as well as new verbs recently introduced officially into the English Language. The idea behind such tests is that it is possible for humans to label an unknown word as a verb, if this is used as a one (at least in a classical transitive way), just by the context and how it is used in a sentence. We learn this by experiencing many times various sentence with similar patterns. I argue that an SRL system should be able to do the same.

Table 2 shows the main tests adopted for Predicate Identification.

4.2 Challenging Arguments classification

Arguments classification is the arguably the hardest and the most important among all SRL sub-tasks, and it involves many capabilities that the model should have learned during training phase. Overall, it is usually treated as a multi class token classification and the difficulty comes exactly from the number of various arguments (classes) that every toke could belong. *Vocabulary or POS* knowledge is for sure needed while identifying various arguments; in order to investigate it, we test the ability to deal with First Names as well personal pronouns. The assumption behind this test is that whichever specific First name appears in the text, it should not change its role in the sentence. In order to pass the test the system needs to recognise the given name (of person name in our case) and predict the correct argument. The same test can be done with pronouns as well, which might be slightly harder as the latter are often uncased but at the same time much more represented in the training data. I propose two **MFT** tests template-based where the role set is consistent (thanks to a template) and many different names and pronouns are tested. Specifically, the ARG0 and the ARGM-COM needs to be identified above all the names tested. For the first test (First Names) it could be argued that this also represents a test for NER capability. Again, this might be correct but as the recognition is only one part of the full classification process, this test will be placed under *Vocab+POS*.

Robustness will be testes similarity as before thanks to typo's perturbation, this time thought, the typo could appear in every point of the sentence. Moreover, I propose a set of Robustness tests with increasing number of perturbation performed on the same sentence. This could give an insight on the extent of the robustness of these system.

For SRL, another capability is hereby introduced called PP-Attachment Ambiguity. This is in reality a well known challenge for syntactic parsers and it was first introduced in (Ratnaparkhi et al., 1994). The ambiguity occurs when a Prepositional Phrase (PP) is attached into sequences like V-NP-PP. The PP might thus be referring to the verb (a VP-*attachment*) or to the noun (NP-*attachment*). An example is: *"The cop killed the man with a gun"*. Here, the PP ([*a gun*]) could both be carry-

Capability	**Name**	**Type**	**Description**	**Example and Expected behavior**
Grammar	Contractions	[**INV**]	Contracting/Expanding Verb forms (often auxiliarys) Prediction should not change	”I **mustn**’t lose my temper” =”I **must** not lose my temper”
	Irregular Inflections	[**MFT**]	Verbs that present irregular inflections when in past tenses	”Sarah Beheld an Old Rose.” [V:Beheld]
Robustness	Typos	[**MFT**]	Swap one character of the verb with its neighbor	”They cathc a great one” [V:cathc]
Ambiguity	Polysemy	[**DIR**]	Some words with multiple meanings, one of each is a verb.	”They are going to a play at the theater.” != ”Let’s play a game of basketball.” *play* not in verbs != *Play* as a verb
	Gerunds	[**MFT**]	Some gerunds appear as nouns or adjective but they still introduce a roleset.	’What is your **reading** on the situation?’ [V:reading]
Rarity	Slangs	[**MFT**]	Verbs that occur in English slang identification	”I wanna go to the movies tonight.” [V:wanna]
	New verbs	[**MFT**]	Verbs that only recently appeared in the language vocabulary	”He ghosted me after our first date and never replied to my messages.” [V:ghosted]

Table 2: Tests for Predicate Identification

ing information about the action (the killing was executed with a gun) or about the noun (the man had a gun). There is no conventional rule to solve this issue and it is sometimes source of ambiguity for humans as well. This is although rare as we are usually able to untangle the ambiguity with the help of common sense and thanks to knowledge about the entities in the PP. For this reason, this test could also be placed under the *Taxonomy* capability tests, as it requires information about classes and relation of the entity at play; for example: *”I ate a chicken with hands”*, it is obvious for us that chickens don’t have hands while eating is an action that can be done with hands. Two tests are proposed: one is a *DIR* test where the PP is even introduced by the same preposition but the name inside is different (which should trigger a change of argument class), and the second is an *MFT* based on a large dataset available online where all the PP refer to the noun and need to be properly classified. More details about this test can be found in the test implementation paragraph below.

When it comes to Span Identification two tests are proposed that challenge the model to be able to identify the group of token as a signle argument: in the first case because it one long Person name (MultiSpan Entities), and the second is a long span mostly constituted by adjectives which no matter theyr number of their sentiment, should not change the semantic role of what they refer to with respect to the predicate.

Finally, for *Paraphrasing*, an Active/Passive transformation test is implemented to check whether a system is able correctly identify (before and after) what is the Agent and what is the Patient in the action. This is particularly interesting as in the passive voice, The Agent does not corresponds with the grammatical subject anymore (same for Patient and object).

5 Data Generation and testing

As the tests presented in this work aim at evaluating the performances in the real world, no traditional data set was entirely adopted. As state already, using common benchmarks along with annotated labels can be useful for a fair comparison of various approaches and it definitely provides a quick and easy way to evaluate such as conventional metrics and performance scores but on the other hand, if no further investigations are conducted, it is hard to understand in details the capabilities of a model on real world situations and peculiar inputs. For this reason it is always a good idea to undergo some stress-tests with carefully planned data that should be put together specifically for the occasion data. This can be done manually or with the help of tools such the above mentioned Checklist. Golden labels for the data should also be generated. In this paragraph the main generation methods will be discussed, followed by some specific test implementation details. The number of instances generated per test can be found in Table 4 and Table 5 as well as in the Appendix, where for every test the methods and a description is provided (Table 6 and 7).

5.1 Synthetising Language

The data set presented was generated in various ways and with the help of different sources. The tests were implemented each with a different structure based on the generated data and the capability of interest. Some of the instances and especially all the labels have been produced manually. One tool utilized was Checklists, which allows to create a large amount of sentences automatically and it further allows to add perturbations such as Typos and Contractions. Moreover, this sentences will have the same set of labels, reducing the manual annotation work to basically none. For instance, while testing *Vocab+POS* in AC, a tem-

Capability	**Name**	**Type**	**Description**	**Example and Expected behavior**
Vocabulary+POS	First Names	[**MFT**]	First Names are to be labeled	*Pamela went with Christopher to the theatre.* — **ARG0**:Pamela, **ARGM-COM**:with Christopher
	Pronouns	[**MFT**]	Pronouns to be labled correctly.	*He went with us to the memorial* — **ARG0**:He, **ARGM-COM**:with us
Robustness	Typos	[**MFT**]	Swap one character of the verb with its neighbor. Typos are added up to 4 times.	”Jason and I had a great conversation.” = ”Jason and I had[a] great [oc]nve[sr]ation.” In here 3 typos.
PP-attachment Ambiguity/Taxonomy	PP-Attachment disambiguation	[**INV**]	PP-Attachment correct labeling	*I fixed the car with a red logo* $\neq$ *I fixed the car with a hammer*
	PP-Attachment refers to noun	[**MFT**]	Large set of instances all with PP-A. referring to the noun is tested	*receiving number of approaches* ARG-1
NER / Span Identification	MultiSpan Entites	[**MFT**]	Long Named entities should be identified in the same span	[Niels Minh Abanda] saw [Said Jacob Sugiyama] in [Asella]”
Span Identification	Long spans	[**MFT**]	Long span constructed with a list of adjectives to be detected.	*[The Christian Bhutanese homosexual friend of mine] is well loved* — first six words are the same argument.
Paraphrasing	Active/Passive	[**INV**]	The arguments are supposed to remain the same. Arg0 and Arg1 labes are checked.	”[The lawyer] defended [her client] against charges of fraud and embezzlement.” = ”[The client] was defended against charges of fraud and embezzlement by [the lawyer].” [All arguments stays the same]

Table 3: Tests proposed for Argument Classification

plate was created with the use of the frame [*go.02 - self-directed motion, disappear or go away*] as follows: ”{first_name} went with {first_name2} to the mask”. This sentence will always have the same roles for the easy to identify predicate *went*, namely ARG0, ARGM-COM and ARG4. A template was also used for Span Identification thanks to the lexicon from the library, which allows to have a large number of First name from various different languages. Unfortunately, a template based approach leaves sometimes little room for major personalisation. For this reason the functionality of Checklist were sometimes enhanced by some manual work or with free available text found on the open web. A good example of this is the Verb Contractions test (PI-Grammar-Contractions), where a number of sentences (which verb was contractible or expandable) were written manually for checklist to create the twin sentence but with the contracted verb[1], or further the PI-Grammar-IrregularInflections, where a list of irregular verbs was manually scraped from the internet and given to Checklist template. Another big source for sentence generation was ChatGPT (OpenAI, 2023). This is the case of PI-Ambiguity-Polysemy. The well known chatbot based on a pre-trained large language models. Its ability to manipulate language and text were useful for the automatic generation of sentences. Obviously, the output was manually reviewed and often adjusted in case of error or inaccuracy. In the Appendix sections at the end of the document, all the propmts used to synthesizing sentences are reported. For more specific behaviors, exceptions and their golden labels, PropBank[2] was also regularly consulted. For Gerunds test for example, PropBank was foundamental as it provides the part of speech in which the frame could appear in the text. Besides, for PP-Attachment-Ambiguity MFT, a part of an existing dataset was used. Such dataset is introduced in (Ratnaparkhi and Kumar, 2021) by the author to tackle this same specific problem. All the sentences presented follow the same structure of 4 words: Verb, Noun and the attached PP. Hereby, it was adopted to the task by utilizing only the PPs that were referring to the NP and thus labeling them as I-ARG1 I-ARG1. Only the test split[3] was used. Data is available on GitHub[4] as well as all the scripts to generated them. Finally, for PP-Attachment Ambiguity, on external dataset was adopted[5].

5.2 Tests implementation

When implementing any kind of test, is necessary to compare the predictions against the golden labels. Only afterwards, if two predictions are presented together, they can be compared as and Invariance or a Directional test. With respect to predicate identification it is actually not necessary to have all the labels for the full sentence. Often, the index of the predicate of interest can be used to make sure it is identified, for example by looking whether the word at that specific position results in the list of ”verbs” identified. For argument classification, it is not mandatory to verify the full sentence label set

[1]Inspiration for possible forms was actually found in the Checklist *perturbator.py* source code, from `https://github.com/marcotcr/checklist/blob/master/checklist/perturb.py`, line 350 and line 407

[2]`https://propbank.github.io/v3.4.0/frames/`

[3]`https://github.com/adwaitratnaparkhi/ppa_transformer/blob/main/data/RRR1994/PPAttachData/test`

[4]`https://github.com/GabHoo/Challenging-SRL`

[5]`https://github.com/adwaitratnaparkhi/ppa_transformer`

for every instance (as this could actually introduce some noise in the automatic evaluation procedure) but is rather comparing a specific subset to make sure selected behavior is being tested. For PP Attachment MFT test, as all the instances selected are known to contain a PP that refers to the noun. In practice, this means that the PP will have the same tags as the noun. As the template of the sentence is always the same namely 4 words, the lables predicted for the 3rd and 4th words (the PP) are checked against the 2nd (the noun), if they are the same, the amiguity is solved correctly. This can be checked easly thanks to the BIO tags, which should then be in the shape [B-*arg* I-*arg* I-*arg*]. Similar methods was used for long span detection as well, where given the template, the index of the words supposed to be in the same span is known and the equivalence of their prediction is tested.

6 Experiment

Two models are tested against the synthetic test set proposed above. For every different test, the failure rate is reported (reflecting *Checklist* evaluation metric), which is simply the how many times a system did not attend the expected behavior over the total number of instances. As no particular randomness is involved, no repetitions to study statistical significance were performed. Scores are reported separately for PI and AC in table 4 and 5.

6.1 The players

-**Simple BERT** is a model proposed in (Shi and Lin, 2019) which is based on BERT (Devlin et al., 2019) and is fine tuned for relation extraction and Semantic Role Labeling. For SRL specifically, the authors address predicate sense disambiguation and argument classification. Both tasks are achieved by adding a one-hidden-layer MLP on top of the classic BERT architecture. The implementation provided by *AllenNLP* however, does not provide the predicate sense label as output so it can only be used for predicate identification (labeled with ”V”) and arguments identification and classification.

-The second model tested in this work is called **RnnOIE** and is proposed in (Stanovsky et al., 2018) as a bi-LSTM transducer for Open-IE. It is greatly inspired by (He et al., 2017) which introduce such a bi-LSTM structured model achieving for the time SOTA results in SRL taks. Specifically, RnnOIE creates a feature vector for every token by concatenating the embedding of the word, the embedding of its part of speech as well as the embedded head of the predicate and its POS. Duplicating the predicate’s feature allow the model to make predicate-specific word label predictions. Such features are then fed to bi-directional deep LSTM transducer which computes contextualized embedding. The latter are fed to a Softmax to produce a probability over all possible argument tags.

Both model adopt the BIO tagging notation and the span based argument identification. This way, a whole span of words can be labeled as the same argument.

7 Analysis and Discussion

Overall both models reveal numerous failure in some of their core capabilities. Reasonably enough, as supported by the literature (Shi and Lin, 2019), SimpleBERT achieves better results in almost all the test apart from Irregular Inflections, the MFT for PP-Attachment ambiguity and (only by one point) the 4 perturbation Robustness test. Strangely, for Predicate identification, the results of the model are exactly the same. On one hand this can mean that no major improvements have been recorded in the task of predicate identification from one model to the other. On the other hand, this also reveals a big limitation of the test set presented for PI because goes to show a big difficulty unbalance between the instances within the test. To achieve the same percentages of failure, the model probably failed on the same sentences and succeeded on the others, entailing that some were just too hard for both model to solve. For Argument classification, some interesting results can be seen in Table 5. For *Vocabulary+POS*, the first model achieve the perfect score while the BiLSTM based shows a 35% of failure rate on the First Name. This confirms that, on contrary to Predicate Identification, the Argument Classification has improved a lot from one model to the other. RnnOIE for this test, fails to label fairly simple sentences such as *Anne went with Emily to the dance*, predicting *with Emily* as [’B-ARGM-MNR’, ’I-ARGM-MNR’] and not modifier (Company). Interesting are the PP-Attachment Ambiguity tests: in the first (INV), SimpleBERT scores much better, showing better taxonomic knowledge. This test howver was manually done and only has 6 instances. In the MFT test on the other hand, on a large number of instances, BiLSTM performs slightly better. This is not expected as the SimpleBERT should

Capability	Name	Type	Nr. Inst	Bert Model	Lstm Model
Grammar	Contractions	[**INV**]	18	0%	0.0%
	Irregular Inflections	[**MFT**]	35	1.8%	1.8%
Robustness	Typos	[**MFT**]	53	1.8%	1.8
Ambiguity	Polysemy	[**DIR**]	29	3.4%	3.4%
	Gerunds	[**MFT**]	15	73%	73%
Rarity	Slangs	[**MFT**]	12	83.3%	83.3%
	New verbs	[**MFT**]	10	10%	10%

Table 4: Failure rate (%) for Predicate Identification

Capability	Name	Type	Nr. Inst	Bert Model	Lstm Model
Vocabulary+POS	First Names	[**MFT**]	100	0.0%	35%
	Pronouns	[**INV**]	100	0.0%	3%
Robustness	1 Typos	[**INV**]	100	14%	16%
	2 Typos	[**INV**]	100	32%	38%
	3 Typos	[**INV**]	100	47%	47%
	4 Typos	[**INV**]	100	53%	52%
PP-A. A. /Taxonomy	PP-A disambiguation	[**INV**]	6	50%	100%
	PP-A refers to noun	[**MFT**]	1641	22%	19%
NER / Span Identification	MultiSpan Entites	[**MFT**]	100	46%	58%
Span Identification	Long spans	[**MFT**]	30	0.0%	0.0%
Paraphrasing	Active/Passive	[**INV**]	20	0%	5%

Table 5: Failure rate (%) for Argument Classification

perform better thanks to its architecture and the BERT embedding (thought to perform better than the GLOVE initialized embedding of the Bilistm system). The multiSpan ENtity test reveals that both models struggle with long and complex First Names from different languages, this is reasonable as such names barely appeared in the training data and when they come in group it might also be hard to get any syntactical insight from the surrounding words. Overall, the biggest limitations for the task of creating a challenging test set is the lack of labled data, leading to a small number of instances when creating the tests manually. Moreover, most of the sentences only feature one predicate only, and despite the fact that SRL works at a sentence level, it is often the case in the real world many rolesets appear in the same sentence and thus in the same input for the system.

7.1 Future work

When desinging a Natural language processing system for a specific domain is necessary to take into account the related lexicon of the domain. If only a small amount of labeled of data is given, along with a larger unlabeled corpus, developing an evaluation set might not be a trivial task. First off, if the labeled data is enough, a good idea might always be training a system with it and then let the unlabeled data go trought a forward pass of the model; if the predictions are any reasonable, it might be worth checking manually the labels given by the model and correcting them if necessary. This is simply because manually controlling somehow decent prediction is much faster way to obtain labeled data then manually annotating from scratch. That aside, for a test suits that takes into consideration the domain difference, it would be appropriate to first off focus on the frames (rolesets) specific to that domain. Identify as many as possible from the labeled data and find them in unlabeled corpus. Can those be identified? Moreover, if in posses of knowledge/lexicon, it would be good to change the arguments of such frames with synonyms. This could be useful for testing Vocabulary. Moreover, it could be also useful to test that same predicate both with inflections form as well as in a rephrased context where arguments are the same but the construction of the period is different. This is because, specifically in a different domain, predicates classification is crucial to then better identify its arguments. Moreover, Ambiguity could also be tested in a different domain, this requires in depth knowledge to identify when and how expressions could be ambiguous but, if solved by the system, it would show great Vocabulary capability.

8 Conclusions

In this work, after arguing in favour of the importance of a challenge data set and explaining Checklist most important standpoints and methodoligies, I identified some of the most important capability that every Semantic Role Labeling system should have and furthered introduced possible ways to test them. A test set was then created either manually or with the help of third party software such as chatGPT or the Checklist software tool. Two models were tested, one based on BiLSTM architecture and another one on the famous transformer-based encoder BERT. Results show how this struggles in some of the core tests proposed such as properly identify and classify First names in case of Long Spanned named entities or resolve PP-attachment ambiguity.

References

Olga Babko-Malaya. 2005. Propbank annotation guidelines. *URL: http://verbs. colorado. edu*.

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*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Daniel Gildea and Daniel Jurafsky. 2000. Automatic labeling of semantic roles. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 512–520, Hong Kong. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

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

Sabine Lehmann, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Herve Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold. 1996. TSNLP - test suites for natural language processing. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330.

OpenAI. 2023. Chat with ai - chatgpt.

Martha Palmer, Claire Bonial, and Jena Hwang. 2017. 315VerbNet: Capturing English Verb Behavior, Meaning, and Usage. In *The Oxford Handbook of Cognitive Science*. Oxford University Press.

Adwait Ratnaparkhi and Atul Kumar. 2021. Resolving prepositional phrase attachment ambiguities with contextualized word embeddings. In *Proceedings of ICON-2021: 18th International Conference on Natural Language Processing*, National Institute of Technology, Silchar, India.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

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

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895, New Orleans, Louisiana. Association for Computational Linguistics.

A Appendices

A.1 PROMPT for automatic data generation

Some example of propmpts used to obtain data from chatgpt.

-Test [PI-Amibiguity-Polysemy]: *Some [words] have multiple meaning, specifically I need [words] that can be verbs but also can be used in other contexts as a nouns or adjectives. Some examples are for inspiration: [Dress, bank, store, catch, attack, left, challenging, play, worry, fold, mix etcc]. Can you find 40 of these [words] and provide a pair of complex sentences where such [word] is used first as noun or adjective and in the second sentence as a verb? The sentencee should be as similar as possible. It should be a json format where the key of the dictionary is the verb and the value is the list of the two sentences. Minimum 30 instances*

-TEST [PI-Robustness-Typos]: *Can you provide a list of 50 verbs in present OR past tenses (not both) and put them in a python list? They should be transitive verb*

-TEST[PI-Rarity-Slang]:*This list contains slangs verbs or abbreviations, can you provide an exmaple sentence for all of them? [wanna,gonna,"gotta","gimmie","lemme","dunno", "tryna","Ima","Needa","Hafta","Whatcha","Cmon"] Create a python dictionary if possible where the key is the provided word and the value is the sentence*

Capability	Name	Type	Generation Methods	Description
Grammar	Contractions	[**INV**]	Manual+Checklist	A list of of sentences were manually written and Checklist created its duplicate contracted or expanded
	Irregular Inflections	[**MFT**]	Scraped + Checklist	List of inflected irregular verbs was found online and Check list generated the sentences
Robustness	Typos	[**MFT**]	Chatgpt + Checklist	ChatGPT provided a list of verbs and Checklist perturbated and created sentences with them.
Ambiguity	Polysemy	[**DIR**]	Manual + ChatGPT	ChatGPT was given some example from the author and provided more. Output was manually checked
	Gerunds	[**MFT**]	ChatGPT+Manually	Same as above
Rarity	Slangs	[**MFT**]	Manual + Scraped +chatGPT	A list of expressions were scraped, given to chat gpt and then manually checked/modified
	New verbs	[**MFT**]	Manual + Scraped + ChatGPT	A list of expressions were scraped, given to chat gpt and then manually checked/modified

Table 6: Methods used to create data for every test for PI

Capability	Name	Type	Generation Methods	Description
Vocabulary+POS	Pronouns	[**INV**]	ChatGPt+Manually	Provided some examples in the prompt and checked and extended the output.
Robustness	Typos	[**INV**]	Checklist	Template from checklist perturbated multiple times
Taxonomy/Ambiguity	PP-Attachment	[**INV**]	manual	
	PP-Attachment	[**MFT**]	Test set taken from github.	Only the PP referring to nouns were adopted.
NER/Span Identification	MultiSpan Entites	[**MFT**]	Checklist	Checklist lexicons provides first and second names from various countries as well as city.
	Long Spans	[**MFT**]	Checklist	
Paraphrasing	Active/Passive	[**INV**]	ChatGPT + Manually	

Table 7: Methods used to create data for every test for AC