Intelligent agents can develop good policies for exploiting large data sets

Jeremy Dohmann

Abstract

I use RL to boost the generalizing power of natural language inference models through the introduction of specially curated "virtual examples" - examples created from the original data SNLI data set [1]. This work builds off of the research conducted in "Analyzing Compositionality-Sensitivity of NLI Models", Nie, et al. (2018) [3] in which novel classes of virtual example were created from the SNLI data set to test the ESIM, introduced in [2], NLI architecture's robustness to compositionality in language - or the ability of an NLP model to understand how different structural organization of the same, or similar words can produce new meanings.

More generically, this paper addresses the difficulty of producing quality virtual examples in the natural language space. Though there exists much theoretical work on mathematically precise methods to transform data for more continuous tasks such as facial recognition [4], there is tremendous difficulty in generating virtual examples capable of meaningfully exploring the space of natural language training examples. This is due in part to the non-continuous nature of language, as our sensitivity to ungrammaticality is much, much higher than say, our sensitivity to recognizing faces under occluded conditions. My analysis of the RL agent's exploration of the data taken from [3] will highlight the difficulty of generating serviceable virtual examples within the NLP domain.

Given an original machine learning task and the introduction of some number of subtasks (in this case, the two "virtual example" classes from [3]), I would like to find a strategy to optimally exploit the added training data to achieve high performance over both the original SNLI task, the two subtasks, and a combination of all the tasks at once. I introduce a reward structure and state space over which I run the REINFORCE algorithm with episodic learning.

This study shows that SOS training examples seemingly contaminate the data set by having an adverse impact on performance over the original SNLI data - their mere presence making the learnability of the original SNLI task much more challenging - and that the "optimal" RL agent performs no better than an agent with a uniform random policy.

That said, over time the agent adaptively learns to avoid SOS examples entirely in achieving maximal validation accuracy for the combined task but suboptimal overall reward within the task defined below, due to plummeting accuracy on the SOS subtask. This result shows that the intelligent agent can, in some sense, perform better than any simple policy for utilizing the SOS and AMod data, and that given a different definition of the reward structure and action space (say one that omits the SOS data entirely - a design decision that this paper provides ample support for making), the eventual policy learned by the agent should be considered optimal.

This work also shows that using a minuscule fraction of the 120,000 training examples provided radically improves performance across nearly all tasks - furthermore, training on all 120,000 examples shows further evidence that achieving mastery on

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

the SOS examples may contaminate the data set and have an adverse impact on performance on the other subtasks.

This study's primary obstacle is the difficulty of defining a good reward structure. It shows that using RL to select training examples far exceeds the performance of the naive approach of using all available data. It is both limited in the narrow scope of learning strategies used (in particular it only uses REINFORCE) and by the non-general form which the state space takes.

Finally, it shows that interesting insights about the structure of the original data can be gleaned by the performance and exploration of an RL agent. Future directions should investigate value function approximation coupled with more linguistically motivated state-action representations.

Problem space and data set

Natural language inference (NLI)

Natural language inference has been identified as a crucial task in NLP for decades, and is viewed by many as being an integral necessary condition to true natural language understanding (NLU). Most NLI tasks are interested in the following problem, given a premise sentence *p* and a hypothesis sentence *p*, determine whether the *p* entails *h*, contradicts *h*, or neither. An example is given below:

p: A person on a horse jumps over a broken down airplane.

h: A person is at a diner , ordering an omelette .

One of the key drawbacks within NLI is that many models have been shown to exploit simple heuristics that have nothing to do with deep understanding and there is concern that given current data sets, it may not be possible to design an architecture that doesn't simply default to un-motivated heuritics. This is especially a problem if we want our models to learn compositionality, that is - the recursive semantics of language, and the way meaning changes when lexical content is unchanged but when structure varies.

While many data sets exist for NLI one of the largest and most commonly used is the Stanford NLI data set. The SNLI data set has over 500,000 training examples, 10,000 validation examples, and 10,000 training examples. The sentence pairs above were taken from the SNLI set.

This study focuses on the Enhanced Sequential Inference Model (ESIM) of Chen, et al. (2017). The ESIM model uses several stacked neural network layers including BiLSTMs, pooling layers, alignment layers, so-called "composition" layers and then several different types of output into a softmax layer.

Virtual examples

Introduction of "virtual examples" used generically to try and boost performance of a model acros some metric via introduction of new training examples, potentially artificially created, and oftentimes designed with the express intention of testing some sort of functionality. For example they can be used to produce artificial rotations of an image to make facial recognition models better at identifying subjects occluded under conditions which the original training data may not assess.

In the NLI space, much work has been dedicated to testing compositionality - or the ability of a model to understand the way structural changes in a sentence may induce different entailment relationships. Nie, et al (2018) introduces two new classes of premise/hypothesis pair derived from the SNLI data set: Subject-Object Swap (SOS) which is formed by swapping the subject and object of a sentence creating a new hypothesis which is contradicted by the original, and Add Adjective Modifier (AMod) which takes an adjective modifier of a noun somewhere in the sentence and moves it to modify a different noun, this creates a new sentence which stands in a neutral position w.r.t. the original (that is, there is neither an entailment, nor a contradiction between the original sentence and the one generated from it). Nie provide over 120,000 training examples of both SOS and AMod with which to boost the SNLI training set, as well.

Nie, et. al's data set suffers some major deficiencies, in particular a huge proportion of the SOS premises are either ungrammatical or semantically unfelicitous, rendering their tags, and therefore their inclusion in the data more generally, rather specious. Consider the first three hypotheses from the test set:

1. Two blond another are hugging one women .
2. A few people in a restaurant setting , juice of them is drinking orange one .
3. Two women hugging just had lunch who and saying goodbye .

Despite the seeming simplicity of generating sentences whose subjects and objects have been swapped, none of the above examples are grammatical, and in fact it is rather difficult to find a single example for which the hypotheses are grammatical. As reference in the abstract - this is representative of the state of synthetic sentence generation in NLP, which stands in stark contrast to the advanced methods available for creating synthetic data in other fields such as image processing. Furthermore, I will show below that the lesser quality of the SOS sentences is

RL agent

Action/state space representation

This work takes a very naive approach to action/state space representation. For simplicity, I processed only the first 20,000 SOS training examples and the first 20,000 AMod examples. The action space is discrete and of size 40,000. An action of index $j \in [0, 20,000)$ indicates choosing the j^{th} AMod example available, an action of index $j \in [20,000, 40,000)$ indicates choosing the $(j - 20,000)^{th}$ SOS example. Similarly the action space is simply a vector of length 40,000 where the index j represents the number of times action j has been chosen so far in the algorithm. The agent chooses 5,000 actions and then reaches a goal state and terminates.

Reward structure

The reward structure for the RL agent has been designed to prioritize grammatical sentences (this is in order to mitigate the problems surrounding the SOS set), it also is designed to prioritize well balanced data sets, and data sets for which bag of words models have a difficult time performing above random. Finally, the goal state evaluates the performance of the ESIM model trained on the new data over the boosted SNLI + SOS + AMod test set, as well as the performance over each of the test sets separately. I use the definitions below:

- D : the original SNLI data
- D' : D boosted by virtual examples
- $ESIM_D(D')$: accuracy of ESIM trained on D over the l D + the virtual examples chosen during an episode (D' test set)
- $ESIM_{D'}(D')$: accuracy of ESIM trained on D' over D + SOS + AMod test sets (D' training set)
- $ESIM_{D'}(D)$: accuracy of previous ESIM model on D
- $ESIM_{D'}(SOS)$: accuracy over the SOS test set
- $ESIM_{D'}(AMod)$: accuracy over AMod test set
- $BoW_{D'}(D')$: accuracy of BoW classifier from Nie, et al. trained on D' over D' training set
- $BoW_{D'}(D)$: accuracy of BoW classifier from Nie, et al. trained on D' over D training set
- $MAJ_{D'}(D')$: accuracy of applying majority label of D' test to the D' training set
- $MAJ_D(D')$: accuracy of applying majority label of D test to the D' training set

In most usages of the symbol given above, the test set D' is the SNLI + SOS + AMod test set, and is therefore the same in every episode. In the case, of $ESIM_D(D')$ D' denotes the result of unioning the 5000 examples chosen by the RL agent with the original SNLI test data - this varies between episodes. Furthermore, when D' appears in a subscript it refers to the training set resulting from

combining the original SNLI training set and the examples chosen by the RL agent - this again varies from episode to episode.

There is an immediate stepwise reward given based on a grammatical score evaluated by a word sequence language model. If the chosen action has already been chosen before the stepwise reward is negated and then added to the cumulative reward - this enables the agent to learn relatively quickly to disprefer duplicate choices. There is a batch reward based on class imbalance and success of a BoW model on the data, and there is a goal state reward based on performance over each individual task and how poorly ESIM performs over the data set chosen.

The reward functions are:

step rewar=**min**$((.5\times$**tanh**$(20 \times LM(p) - 2) + .5)/T,\ (.5 \times$ **tanh**$(20 \times LM(h) - 2) + .5)/T)$
$batch\ reward = (.5 \times batch\_size \times ($**tanh**$(2 * (BoW_D(D')/BoW_{D'}(D)) - 1) + 1)/T)$
$\cdot\ (.5 \times batch\_size \times ($**tanh**$(2 \times (MAJ_D(D')/MAJ_{D'}(D)) - 1) + 1)/T)$
$goal\ reward = (2/3)($**gmean**$([$**tanh**$(2.5 \cdot ESIM_{D'}(D) - 1) + 1],$
mean$([$**tanh**$(2.5 \cdot ESIM_{D'}(SOS) - 1) + 1], [$**tanh**$(2.5 \cdot ESIM_{D'}(AMod) - 1) + 1])))$
$+ (1/3)ESIM_D(D')$

Where **gmean** is the geometric mean and **mean** is the arithmetic mean. Note that equal weight is assigned to SOS and AMod accuracy, and cumulatively, those two are assigned approximately equal weight to total validation accuracy. Finally, $ESIM_D(D')$ is assigned the least importance.

The motivation behind the step reward is very straightforward, each action consists of a premise/hypothesis pair, and in theory the quality of the action should be proportional to the probability assigned by the language model to the worse of the two sentences.

The goal reward is motivated by the notion that if training on the boosted data makes the BoW classifier or the MAJ class classifier more performant, then the actions chosen have introduced imbalance, either in the form of changing the class balance or making the classes more predictable from word-based heuristics alone (which in theory, are predominantly spurious and reflect bias introduced by the data set - not necessarily universal truths about how true inference works in English.)

Finally, the goal state reward reflects three priorities in decreasing order of importance - 1) boosting the training of the original ESIM with the actions chosen should maximize performance over the validation set for the overall task (that is, the SNLI data, plus the data from the two adversaries), 2) the boosted ESIM model should be performant on both the SOS and the AMod task (in the discussion I will demonstrate that the results suggest prioritizing the SOS task is a *bad* design), and 3) the original ESIM model trained only on the SNLI data should perform poorly on the original SNLI test data + the 5000 examples chosen during the episode. The final component of the reward seems a bit opaque but it reflects the desire to have our data set get into "hard to reach" areas of the problem space in order to achieve broad coverage.

The exact constants in the goal rewards were chosen so that the cumulative reward is primarily influenced by the goal state reward, with the batch and step rewards having equal contributions which were on the order of the variance of the goal state reward (that way they could be large enough to influence the outcome somewhat).

Learning algorithm

I use a standard implementation of the REINFORCE algorithm using a neural network with input and output layers of size 40,000 and a hidden layer of size 1024. The output layer is softmaxed and normalized to output a probability distribution over the actions.

Methods

The initial ESIM model $ESIM_D$ was trained over the SNLI training set for 5 epochs. The hyperparameters for training and testing were identical to those used in [2]. In order to evaluate the goal reward, at the end of each episode a clone of the ESIM model resulting from the training above is then trained for an additional 5 epochs on the 5000 examples chosen during the episode.

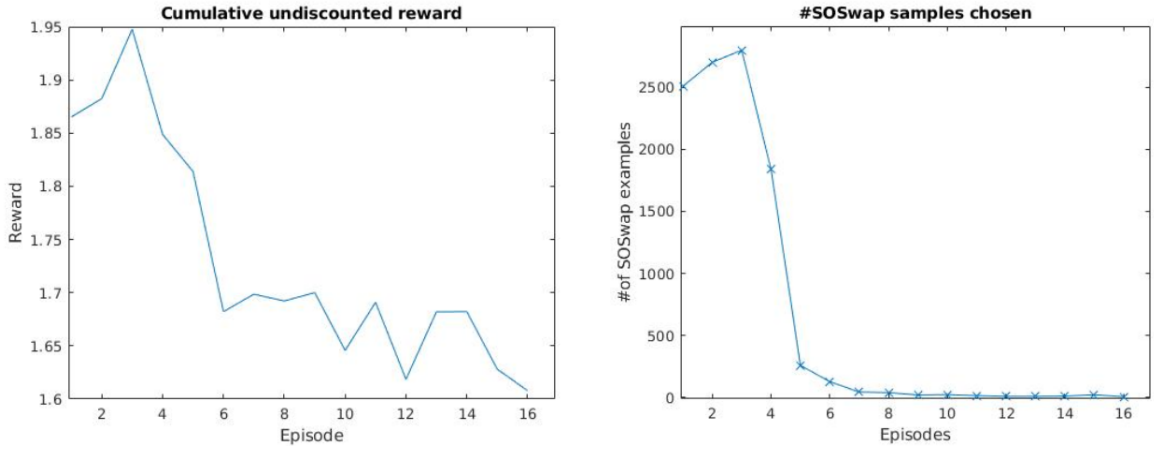
The LM was trained using the pytorch word_sequence_modeling example from the pytorch examples git repository (https://github.com/pytorch/examples). Using an embedding size of 1500 and a hidden layer size of 1500. The model was then configured to take in input sequences word by word and return the average probability assigned to each true word in the sequence.

The BoW classifier converted sentences into one-hot encodings over the vocabulary of the SNLI data set. It had a single layer from the vocabulary size to the number of labels (0,1,2) and was trained over the entire SNLI training set for 10 epochs. I then, at each batch, retrained it on the new batch of training sentences chosen during the batch period.

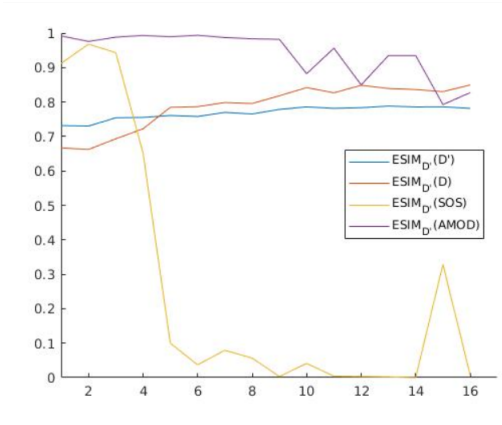
All evaluations of the validation score were performed over the combination of the SOS and AMod dev set and the SNLI dev set. Testing evaluation was performed over the SOS and AMod test set and the SNLI test set.

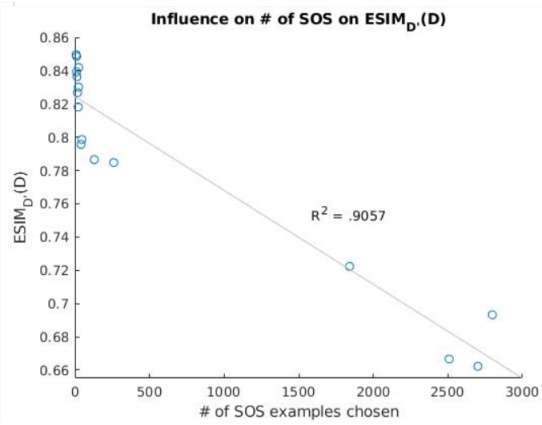
Results

The RL agent peaks in cumulative reward very early on as shown in the graph below. Also shown below, it learns very quickly to not use duplicates, by episode 9 it uses only 5. This is likely due to the penalty for duplicates being assigned immediately - despite it being relatively small compared to the scale of the cumulative reward. Interestingly enough, the number of duplicates chosen by the "optimal" agent (319) is substantially above the expected number of duplicates under a random policy (293.5), and the fairly well performing agent of the previous episode far exceeds that of the random policy (384).



The validation performance $ESIM_{D'}(D')$ improves substantially for awhile and plateaus well after the cumulative reward has begun to fall. This is due to the agent learning to not choose SOS examples which seem to have a strong negative impact on performance over the regular SNLI data. This makes sense as they are mostly of low quality and may be hard to discern from regular SNLI examples.

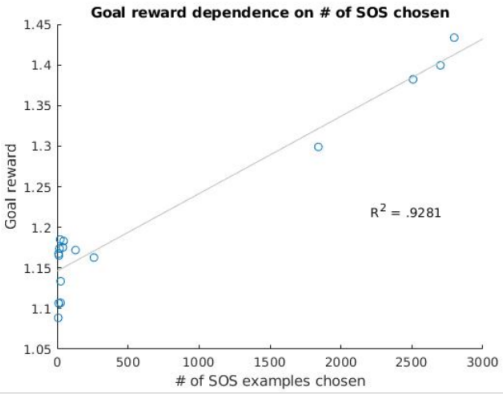


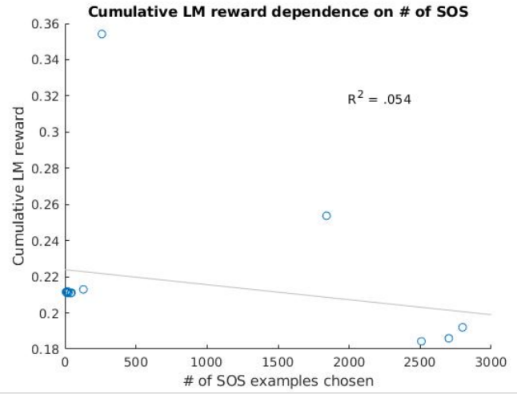


One may think that lower language model scores may cause the RL agent to learn to avoid SOS sentences but, as shown below, there is very weak correlation between cumulative stepwise reward and number of SOS examples chosen. The mean and variance for the two virtual example types are below - though there is a noticeable difference in grammaticality scores between SOS and AMod - the regression showing the weak dependence of cumulative LM reward on # of SOS samples seems to suggest that the LM score difference is an unlikely candidate for encouraging the model to disprefer SOS samples later on in the episodic learning.

	Mean	Variance
SOS	0.051	0.00085
AMod	0.060	0.00094

There is, however, a very strong correlation between the goal reward and the number of SOS examples chosen which is directly related to the fact that the accuracy over the SOS subtask plummets in later episodes as the agent learns to not introduce them into the training data.





Comparison to naive methods

Below is a table showing the performance of various strategies on the SNLI + SOS + AMod test set.

	No virtual examples	All virtual examples	``Optimal" policy	Random policy	Best validation acc policy
ESIM(D')	0.69	0.5	0.73	0.75	0.78
ESIM(SNLI)	0.87	0.54	0.68	0.69	0.85
ESIM(SOS)	0.01	1	0.83	0.94	0.01
ESIM(AMod)	0.08	0	0.99	0.99	0.83

We find that the naive method of using ALL test examples performs poorly across the board. Interestingly enough the "optimal" policy performs slightly worse than a random policy, but the policy learned by later episodes in many ways is more performant - they have overall high validation, high performance on the original SNLI data and high performance on AMod. Performance only lags on SOS which as we've seen is actually a very low quality data set, and therefore has a very good case for exclusion from study entirely.

Note that the results from the second column seem to suggest that inclusion of the SOS examples have a strong negative impact not just on ESIM's ability to learn the original SNLI data but also to even learn the AMod examples. This is very much in line with the subjective analysis given earlier as well as the divergent LM scores between the two types of adversary. This suggests that when all the data is pooled there isn't any discoverable function between inputs and outputs, and that no matter the size or expressiveness or our model it may not be possible at all to get high performance on other tasks. This is sensible as the SOS examples introduce sentence pairs which consistently have no real semantic relationship with one another yet are tagged as being contradictions of one another. There is no real pattern relating SOS premises to hypotheses besides the fact that they have the same words but jumbled up in a nonsensical order. This may be the direct cause for the collapse of the accuracy over the AMod test set in the second column.

Discussion

As discussed, there is qualitative evidence based on reading the SOS examples, evidence from the differing language model scores between AMod and SOS, and evidence from the "All virtual examples" testing data that the SOS data is a contaminant which does nothing besides inject indiscernible noise into the problem space. It is incredibly encouraging that the RL agent learns over time to omit SOS examples entirely. It is slightly disappointing, however, that the resulting decision to continually discount SOS examples represents a suboptimal policy within the reward structure given - making it somewhat unclear why the RL agent learns such a policy.

Based on the regression analysis relating the # of SOS samples chosen in an episode and the cumulative LM reward and the validation accuracy, it seems likely that it is not ungrammaticality which triggers the SOS sentences to be disprefered, but rather, that incrementally decreasing the # of SOS samples chosen increases the accuracy over the SNLI portion of the test set substantially. The gradient descent REINFORCE algorithm therefore likely falls down a "suboptimal" path of phasing out SOS in order to increase the SNLI accuracy.

The fact that the agent learns to phase out SOS and ends up achieving a very good policy over the remaining two categories, provides intriguing and convincing supporting evidence that the SOS data is unsuitable for this task. This is an incredibly important conclusion, as it is not at all trivial to generate good synthetic natural language data let alone quantitatively decide which virtual examples are high quality and which are low quality. The fact that three separate pieces of evidence point to the SOS set being low quality and that the RL agent learns this fact *as well* bodes well for the efficacy of using RL agents to explore optimal usage of natural language training data. This is especially so because the pieces of evidence used to suggest that the SOS data contaminates the other subtasks is far from conclusive, and the RL agent discovers for us that excluding SOS altogether proves, in some sense, to be a prudent usage of the data provided.

Thus, though it is lamentable that the "optimal" RL agent doesn't perform much better than the random policy, I think those results can mostly be attributed to a reward structure which erroneously gives any consideration to the SOS task whatsoever and that rerunning the experiments without the SOS data at all may show that the RL agent is capable of learning truly superior exploitation of the data sets provided.

Conclusion

This paper shows that using RL agents to choose the best subsets of a training set has great promise within the field of NLP. This is an especially useful conclusion given the inherent difficulties of obtaining and producing high quality natural language training data. This specific study finds that by boosting the SNLI data set [1] with the virtual examples in [3] it is possible to train [2] to get very high performance on both the AMod subtask and the original SNLI data but likely not possible to

simultaneously obtain high accuracy on all three of the SOS, AMod, and SNLI data sets due to the contaminating nature of the SOS data. The RL framework used in this example study was somewhat deficient in the large parametrization required for the space. Rather than using the REINFORCE algorithm it may be advantageous to use value function approximation as well as intelligent, more linguistically motivated representations than the state and action space besides just a one-hot encoding of the examples chosen so far. Doing so would not only decrease the size of the parameter space and make the learning process conclude more quickly, but it would also provide the potential for transfer learning from one data selection task to another. Furthermore, representing actions via continuous, embedding-like representations and learning a policy via value function approximation would grant the RL agent the opportunity to more directly compare the similarity between different actions in the space by leveraging representations that encode semantic relatedness as opposed to merely latently learning the interactions between the 40,000 or so possible actions via the relatively few episodes it had to learn from.

One of the primary time constraints of the learning process was the massive scale of the ESIM architecture and the overhead of evaluating it and training it at the end of each episode. It may be possible to use a much, much smaller architecture during the training process to boost the number of possible training episodes available to our RL agent and still end up achieving a policy which is optimal for both the reduced-sized architecture used to train the policy and for some larger architecture of interest such as ESIM. In general, another area of interest within this space would be to study the ability of using one architecture in the goal reward for the RL agent and another architecture in the evaluation. In theory, the optimal data set, of those selectable from the entire action space, should be generically high quality no matter what architecture is used to evaluate it, so it should be possible to use one architecture as a proxy for another during training of the policy.

Another, very important point to note is that a random policy that used 5000 examples was much better than the policy which used all 120,000 - this is indicative that its not just the exact samples chosen, but the number of samples chosen which impacts the quality of the results. I hand chose a finite time horizon of 5000 merely to make the training process go more smoothly - an interesting area to explore would be to permit the agent to have a flexible time horizon and decide on its own when it should terminate the search, as there is nothing special about the number 5000 and it could very well be the case that some other number may get far better performance. That said - the fact that the agent at the end achieved accuracy over the SNLI data set only 2 percentage points worse than the regular ESIM benchmark and similarly good accuracy on the AMod set shows that there may not be much room for improvement with the ESIM architecture and that the policy learned by the end of the 16 episodes may truly be approximately as good as it could get, no matter what time horizon we have.

References

[1] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015.

[2] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, and Hui Jiang. Enhancing and combining sequential and tree LSTM for natural language inference. *CoRR*, abs/1609.06038, 2016.

[3] Yixin Nie, Yicheng Wang, and Mohit Bansal. Analyzing compositionality-sensitivity of NLI models. *CoRR*, abs/1811.07033, 2018.

[4] Partha Niyogi, Federico Girosi, and Tomaso Poggio. Incorporating prior information in machine learning by creating virtual examples. *Intelligent Signal Processing*, 2009.