An Automated and Exhaustive Natural Language Inference Corpus

1 Introduction

The field of *natural language inference* concerns the problem of determining whether a hypothesis sentence h can be inferred by premise sentence p. This problem is essential for understanding natural language and is related to tasks such as retrieving semantic information. Approaches to NLI in the past have included symbolic logic, knowledge databases, and neural networks.

Our objective here is to provide a large scale NLI corpus for the evaluation of learning models that exhaustively explores a precisely defined set of potential examples. We present here the Automated and Exhaustive NLI corpus (AENLI) which contains 1,245,995 automatically generated labeled pairs of sentences, making it slightly more than double the size of SNLI and MultiNLI which are two order of magnitudes larger than existing corpora. All premise and hypothesis sentences consist of independent clauses that contain a transitive verb, an optional adverb, a subject that is an agent, an object that is a physical thing and are possibly negated or passive. These independent clauses are conjoined using *and*, *or*, and $if...then$. The structure of the premise and hypothesis sentences are determined independently and randomly, meaning that every possible sentence pair of this form is generated and this set of potential examples is exhaustively explored.

2 Related Work

There are many existing NLI corpora available for public use. A significant source of NLI data has been the Recognizing Textual Entailment (RTE) challenge tasks. This data is hand labeled and high quality, but suffers from the small size of less than one thousand examples. The Sentences Involving Compositional Knowledge (SICK) corpus for the SemEval 2014 task has a few thousand examples and is semi-automatically generated (Marelli et al. 2014a). Recently, the SNLI and MultiNLI corpora were created using Amazon Mechanical Turks and consist of hundreds of thousands of human generated and labeled sentences (Bowman et al. 2015, Williams et al. 2017).

The SNLI and MultiNLI corpora were created to address perceived drawbacks of existing NLI corpora, namely that they are too small for modern models requiring large amounts of data, they include automatically generated data, and they suffer from indeterminacies of event and entity coreference that harm annotation quality. The AENLI corpus addresses the problem of size and indeterminacies through an automatic generation process that can produce a large amount of data and follows precisely defined assumptions that remove all indeterminacies.

However, the AENLI corpus obviously does not address the perceived drawback of automatically generated data, as it is completely automatically generated. Clearly the AENLI corpus can be perfectly classified using a AENLI corpus specific model, as such a model was used to automatically generate labels for the examples. However, this does not invalidate it as a useful testing ground for general purpose NLI models, as the models evaluated here do not perform close to perfectly. Additionally, the fact that AENLI corpus is automatically generated allows for a precise understanding of what is being learned and revealing in depth error analysis.

3 Data Generation

The AENLI corpus is an automatically generated NLI dataset. Each example in the AENLI corpus consists of a premise sentence P, a hypothesis sentence H, and a label L where the label L is either entails, contradicts, or permits.

3.1 Generating Cores

The first step of generating the AENLI corpus was the hand generation the sets S_A, S_O, and S_V where S_A is a set of nouns that are agents, S_O is a set of nouns that are physical things, and S_V is a set of verbs that coherently can be used with an agent subject and physical thing object. Let C be a core if for some $A \in S_A$, $O \in S_O$, and $V \in S_V$ we have $C = \langle A, V, O \rangle$. For example, the following are cores $\langle man, rub, tree \rangle$, $\langle firefighter, lick, car \rangle$, and $\langle father, hug, rock \rangle$. The cores are generated using using Algorithm 1 below, and the result is a set $Cores$ where $|Cores| = |S_A| * |S_O| * |S_V|$.

3.2 Determiners and Negation

Now we will describe how cores are used to create fully formed English sentences that can serve as hypotheses and premises. Let $\mathcal{D} = \{a, the, every\}$ be the set of determiners that will be used in premises and hypotheses. Moving forward, it will be useful to

Algorithm 1 Core Generation

```
Cores = ∅
for A ∈ S_A do
    for V ∈ S_V do
        for O ∈ S_O do
            Cores ← Cores ⋃ ⟨A, V, O⟩
```

think of a as an existential quantifier and *every* as a universal quantifier. Given a core $C_1 = \langle A_1, V_1, O_1 \rangle$ an example of the following form can be generated where D_1^P, D_2^P, D_1^H and D_2^H are randomly selected from $\mathcal{D}$ and the presence of negation is randomly determined.

Premise: D_1^P A_1 (does not) V_1 D_2^P O_1

Label:L

Hypothesis: D_1^H A_1 (does not) V_1 D_2^H O_1

The scope of negation and the quantifiers must be set so every sentence has a fixed interpretation. For all examples in the AENLI corpus, scope follows directly from surface level syntax, and so the negation scopes over the second determiner and the first determiner scopes over both negation and the second determiner. This means that the interpretation of the deeply ambiguous sentence *A man does not kick every ball* is fixed to have the interpretation that there exists some man and there is some ball that man doesn't kick.

We will also assume that if a noun is mentioned, there is at least one of those nouns. This has implications for the meaning of *every*, so the sentence *Every man kicks a ball* cannot be interpreted as being vacuously true when there are no men to refer to. A final assumption is made regarding entity coreference. If two NPs have the same noun and the determiner *the* then it is assumed the two NPs refer to the same entity. However, the same is not assumed for the determiner a. The first example below highlights assumptions made about *every* and *the* and the second example highlights assumptions made about a.

Premise: Every man kicks the ball

Label:Entails

Hypothesis: A man kicks the ball

Premise: A man kicks the ball

Label:Permits

Hypothesis: A man does not kick the ball.

With our assumptions made clear, examples have unambiguous premises and hypotheses and therefore a single correct label. Now let us present a way of determining the value of the label L for any given determiners D_1^P, D_2^P, D_1^H, and D_2^H. There are three cases to consider: the case where neither the premise nor the hypothesis is negated, the case where only one of the two are negated, and the case where both are negated.

Begin with the first case, where neither the premise nor the hypothesis is negated. It is obvious that such a case can only be labeled entails or permits. The relation $>_S$ on $\mathcal{D} \times \mathcal{D}$ is defined in Figure 1 to capture the how a determiner affects the strength of the statement it is in. Notice that the label for an example of this type should be entails if and only if $D_1^P >_S D_1^H$ and $D_2^P >_S D_2^H$. This makes sense when our understanding of $>_S$ is considered, as the premise of an entailment is stronger than its hypothesis.

	every	the	a
every	X	X	X
the		X	X
a			X

Figure 1: A definition of the relation $>_S$: If an X is present at $(D_{\text{row}}, D_{\text{col}})$ then $D_{\text{row}} >_S D_{\text{col}}$

Now let us move on to second case, where both the premise and the hypothesis are negated. It is again obvious that such a case can only be labeled entails or permits. Under our assumptions, the negation in the premise and hypothesis only effects D_2^P and D_2^H. As such, we have a second relation $>_{S'}$ on $\mathcal{D} \times \mathcal{D}$ defined in Figure 2 to capture how a determiner under negation captures the strength of the statement it is in. Notice that the label for an example of this type should be entails if and only if $D_1^P >_S D_1^H$ and $D_2^P >_{S'} D_2^H$.

	every	the	a
every	X		
the	X	X	
a	X	X	X

Figure 2: A definition of the relation $>_S$: If an X is present at $(D_{\text{row}}, D_{\text{col}})$ then $D_{\text{row}} >_S D_{\text{col}}$

Finally the third case, where only one of the premise and hypothesis is negated. It is obvious that such a case can only be labeled contradicts or permits. Because contradiction is symmetric, it does not matter which is negated so for our purposes let

the premise be the negated. The relation $=_O$ on $\mathcal{D} \times \mathcal{D}$ is defined in Figure 3 to capture whether two determiners individually applied to the same noun would result in overlapping domains. Notice that the label for an example of this type should be contradicts if and only if $D_1^P =_O D_1^H$ and $D_2^P >_{S'} D_2^H$. It is unclear why $>_{S'}$ can be used in this way, but this does generate the correct labels.

	every	the	a
every	X	X	X
the	X	X	
a	X		X

Figure 3: A definition of the relation $=_O$: If an X is present at $(D_{\text{row}}, D_{\text{col}})$ then $D_{\text{row}} =_O D_{\text{col}}$

3.3 Passives

Another step in the generation of examples is the randomly making premise and/or hypothesis sentences passive. Passive sentences are typically considered synonymous to their active counter parts, however our assumption that scope follows from surface level syntax results in some sentences changing in meaning when they are made passive. For example, *Every man kicks a ball* is interpreted as each man kicking some ball, but not necessarily the same ball and *A ball is kicked by every man* is interpreted as there being a single ball that is kicked by every man.

A passive premise and a passive hypothesis can be labeled using the same process to label a non-passive premise-hypothesis pair. This means that the only premise-hypothesis pairs that our framework thus far does not address are those where one sentence is passive and does not have an active sentence with equivalent meaning and one sentence is active and does not have a passive sentence with equivalent meaning. Below is a complete list of forms of active and passive sentences that do not have counterparts with equivalent meanings.

1. Every Agent Verbs an Object.
2. An Agent Verbs every Object.
3. Every Agent does not Verb every Object.
4. An Agent does not Verb an Object.
5. An Object is Verbed by Every Agent.
6. Every Object is Verbed by an Agent.
7. Every Object is not Verbed by every Agent.
8. An Object is not Verbed by An Agent.

If the two sentences in a premise-hypothesis pair one sentence is of a form in 1-4 and the other is of a form in 5-8, the correct label will be permits except for two cases where the label will be contradicts. The first case is where one sentence is of form 7 and one sentence is of form 2. The second case is where one sentence is of form 4 and one sentence is of form 5.

3.4 Adverbs

The next step of data generation is the optional placement of an adverb in a random location within sentence. Adverbs in front of the main verb or at the end of the sentence, call these verb adverbs, result in the same meaning, while adverbs in front of negation, call these negation adverbs, result in a different meaning. If a premise-hypothesis pair would already be labeled as permits, the addition of an adverb to either or both sentences would not change this label. If it would not be labeled as permits, the addition of an adverb could have no effect or change the label to permits.

If an example has the label entails and sentences without negation, with adverbs the label remains the same unless only the hypothesis has an adverb. If an example has the label entails and sentences with negation, with adverbs the label remains the same unless only the hypothesis has a negation adverb or the premise has a verb adverb and the hypothesis has no adverb or a negation adverb. If an example has the label contradicts, with adverbs the label remains the same unless the sentence with negation has a verb adverb.

3.5 Introducing Noise

At this point, every example has a premise and hypothesis that share the same nouns, verbs, and adverbs. To increase the diversity of the dataset, new examples are added where the premise and hypothesis are generated using different cores. The label for these examples is always permits.

3.6 Boolean Expansion

The result of the generation process outlined so far is a data set consisting of labeled premise-hypothesis pairs where both sentences are simple sentences. The final step of data generation is to expand the dataset to include premise-hypothesis pairs where both sentences are compound sentences by using simple boolean logic. If we have the labeled examples:

A entails B

C entails D

E contradicts F

G contradicts H

Then we can generate the compound sentence examples:

A and/or C entails B and/or D
E and/or G contradicts F and/or H
If B then C entails If A then D
If B then E entails If A then F
F or B entails If E then A
F or G contradicts If E then H

3.7 Final Dataset

The final result of the data generation process is a training set consisting of 1,245,995 examples and a test and development set consisting of 47,995 examples. Additionally, there is a disjoint test set of 47,995 examples that uses completely different nouns, verbs, and adjectives that the training set.

4 Methods and Experiments

We evaluated the AENLI corpus on three models: a logistic regression model, a sequence to sequence neural network model, and a sequence to sequence neural network model with attention.

The logistic regression is a simple baseline model. The non-lexical features used for the logistic regression model were the BLEU score between the hypothesis and premise using 1,2,3 and 4 length n-grams, the length difference between the hypothesis and premise, and the word overlap between the hypothesis and premise as a percentage and a count for both all words and nouns and verbs. The lexical features used for the logistic regression model were indicators for every word and every pair of words in the premise and hypothesis and indicators for every pair of words across the hypothesis and premise. These features are largely similar to those used to first evaluate the SNLI (Bowman et al. 2015).

The sequence to sequence neural network model makes use of gated recurrent units (GRU) and has the structure of the model in Sutskever et al. 2014. The GRU is a recurrent neural network that is able to learn long term dependencies by introducing a memory mechanism (Cho et al. 2014). The equations for a GRU can be seen below, where x is the input. At each step t, a part of the input, x_t is inputed and an encoding, h_t is outputted. The encoding h_t is a composition of the current input x_t and the previous encoding h_{t-1}, where z_t and r_t determines the nature of this composition. Allowing the information in h_{t-1} to carry on to h_t gives this model memory. The final h_t outputted is called the encoding of x.

$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1})$
$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1})$
$\tilde{h}_t = tanh(Wx_t + r_t \circ U^{(r)}h_{t-1})$
$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$

The input to our model was the premise and hypothesis represented with 300 dimension GloVe vectors (Pennington et al. 2014). The sequence to sequence model encodes the premise into a single vector using an GRU, and then encodes the hypothesis into a single vector using a GRU initialized with the encoding of the premise. Then the encoding of the hypothesis is ran through a 3-way softmax classifier for a final prediction.

The sequence to sequence neural network model with attention has the structure of the first model in Rocktaschel et al 2016. The equations for this attention model are below, where Y is a matrix with columns that are the encodings outputted from a GRU with the premise as the input, and $h_N \bigotimes e_L$ is the final encoding of the hypothesis repeated L times where L is the length of the premise.

$M = tanh(W^yY + W^h h_N \bigotimes e_L)$
$\alpha = softmax(w^T M)$
$r = Y\alpha$
$h^* = tanh(W^p r + W^x h_N)$

This model is an extension of the sequence to sequence model that uses the encoded hypothesis to compute weights, α, for the outputs of the premise ran through the GRU, Y. Then the encoded hypothesis h_N and the weighted encoding of the premise r are combined into a final encoding h^* that is ran through a 3-way softmax classifier for a final prediction.

For the two neural net models, a single layer GRU was used with state size 32 and an Adam optimizer was used during training. A hyperparameter search was performed over drop values [0,0.1,0.2], L2 regularization values of [0, 0.0005, 0.001, 0.005], and learning rates of [0.0005, 0.001, 0.005] and the highest performing parameters were used on the test set. A hyper parameter search was performed on the logistic regression over the same values, except for the drop-out as that would not be appropriate.

5 Results and Discussion

The performance of each model with the metric of accuracy can be seen in figure 4. Each model was trained until performance on the development set no longer improved. As expected, the neural network models perform significantly better on the data than the logistic regression. There seems to have been some over fitting with the logistic regression model, despite a hyper parameter search being run. Also, we can see that the logistic performs worse

on the disjoint test set compared to the joint test set. This is expected, as the train set and disjoint test set consist of entire different nouns, verbs, and adverbs and the logistic regression makes use of indicator features. This means the disjoint test set will have many features that the logistic regression never saw during training and the logistic regression will have trained weights for many features that don't appear any where in the disjoint test set.

There a multiple interesting things to note about these results for neural networks. First we should note the very small difference between the training set performance and the joint test set performance. For the sequence to sequence model there was no difference in performance and for the attention model the performance was 0.3 better on the joint test set (We expect this was an anomaly). This indicates there was no over fitting on the neural network models. This makes sense, as the joint test set and training set were created using the same randomized process and use the same vocabulary so the data should be distributed close to identically. Also, the training set consists of over 1 million examples, and such a large size should reduce over fitting.

Next we should note that there was only a slight performance drop from the joint test set to the disjoint test set. This is impressive, and indicates that the neural networks were able to learn the patterns in the data independent from the nouns, verbs, and adverbs present. This nullifies the reasonable worry that a training set with a vocabulary of only a couple hundred words will produce a model that can only perform well on examples using that vocabulary.

Finally, we should note that the performance here is not close to perfect. As opposed to the SNLI and MultiNLI corpora which have noisy human generated data that a model likely cannot perfectly learn, the AENLI corpus is automatically generated and therefore we know a model can achieve perfect accuracy. This validates AENLI as a challenge corpus.

	Disjoint Test	Joint Test	Train
regression	71.3	78.5	99.1
seq2seq	88.8	89.4	89.4
attention	90.2	91.2	90.9

Figure 4: Accuracy achieved by each model on AENLI training and test sets

We will now perform a more in depth analysis of the performance of the sequence to sequence neural network model. Begin by considering figure 5, which is the confusion matrix for this model on the disjoint and joint test set. The cell at (ROW, COL) indicates the proportion of examples that were labeled ROW and predicted by the model to be COL for the joint/disjoint test sets. We can see that the model rarely mistook entails and contradicts for each other, which makes sense as it would be easy to learn that entails only occurs with sentences that are both negated or both not negated and contradicts only occurs with sentences where one is negated and one is not. Most errors occurred mistaking permits for contradicts or entails and vice versa.

	entails	contradicts	permits
entails	0.173/0.171	0.001/0.001	0.027/0.027
contradicts	0.001/0.001	0.166/0.165	0.030/0.030
permits	0.030/0.031	0.020/0.022	0.550/0.550

Figure 5: Confusion matrix for seq2seq model on the joint/disjoint test sets

We can also analyze how the sequence to sequence model performed on different types of compound sentences and on simple sentences. In figure six we can surprisingly see that the model performed the worst on examples with simple sentences and compound sentences with *and*.

	simple	and	or	if then	if then w/ or
joint	0.89	0.81	0.91	0.91	0.93
disjoint	0.88	0.81	0.90	0.92	0.92

Figure 6: Accuracy of seq2seq model on different types of sentences.

The AENLI offers an interesting new challenge to neural networks. The dataset requires first order logic reasoning at the words level and boolean logic reasoning at the clause level. For this reason, neural network models that attempt to more closely simulate symbolic reasoning may achieve better results on this dataset.

6 Conclusion and Future work

We conclude by presenting the AENLI corpus to the public for the evaluation of models. In the future, we hope to expand the data set to include adjectives as well. We also hope to experiment with more neural network models on the data, particularly those which use syntactic trees.

I would like to acknowledge George Pakapol, Lauri Karttunen, and Ignacio Cases. Some of this code was written in research with them over the summer.

7 References

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014b. A SICK cure for the evaluation of compositional distributional semantic models. In Proc. LREC.

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

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. CoRR abs/1704.05426.

[Rocktaschel et al.2016] Tim Rocktaschel, Edward Grefenstette, Karl Moritz Hermann, Tomas Ko cisk y,‘ and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In Proceedings of the 2016 ICLR, San Juan, Puerto Rico.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation

Sutskever, I., Vinyals, O., and Le, Q. (2014). Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems (NIPS 2014).

K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoderdecoder approaches. arXiv preprint arXiv:1409.1259, 2014.