ETHAN at SemEval-2020 Task 5: Modelling Causal Reasoning in Language using neuro-symbolic cloud computing

Len Yabloko
Next Generation Software,
New City, New York, USA
lenyabloko@gmail.com

Abstract

I present ETHAN: Experimental Testing of Hybrid AI Node implemented entirely on free cloud computing infrastructure. The ultimate goal of this research is to create modular reusable hybrid neuro-symbolic architecture for Artificial Intelligence. As a test case I model natural language comprehension of causal relations from open domain text corpus that combines semi-supervised language model (Huggingface Transformers) with constituency and dependency parsers (Allen Institute for Artificial Intelligence). The experimental results presented in this paper show potential for SOTA-level performance and demonstrate hybrid Artificial Intelligence in action. On SemEval-2020 Task 5 Subtask 1 ETHAN achieves F1 0.878 which is the second best result at the time of my writing. On Subtask 2 ETHAN currently scores 0.673 which is within the top ten results. All code used for experiments is available as open source and can be executed directly from the web browser without any installation or configuration. I briefly discuss some theoretical approaches that led to the proposed solutions and future directions of this research.

1 Introduction

Several hard open problems of machine learning and AI are intrinsically related to causality (**?**). One of them is the algorithmization of *counterfactuals* (Pearl, 2011): what algorithms would allow machines to recognize crucial semantic difference between sentences like

> If Oswald didn't kill Kennedy, someone else did,
>
> If Oswald hadn't killed Kennedy, someone else would have.(Adams, 1975)

What additional knowledge would a machine require to do such a binary classification? But going even further, what linguistic features of the two sentences would need to be isolated and analyzed before any form of causal inference can be performed? If the first question conveniently re-frames the linguistic problem into statistical one, then the second question requires machine to identify some basic causal relation in the text, namely the *antecedent* and *consequent*.

SemEval-2020 Task 5(Yang et al., 2020) asks these two question in a form of subtasks 1 and 2[1]. Even for a single language (English) the task so fundamental to natural language processing (NLP) that it bridges the major divide in the artificial intelligence (AI) research between the good old fashioned AI (GOFAI) and the methodology that led to the latest advances, namely the Deep Learning (DL).

It is worth mentioning the intense debates[2] around the future of AI which inevitably circles back to the role of causality in human and machine understanding of language or any data for that matter. My results add to the position that any successful approach must be a hybrid between symbolic and connectionist sides of the divide (Marcus, 2020) . I was able to achieve a SOTA F1 score 0.878 on the subtask 1 in a short time using open source software. However, I found subtask 2 significantly more challenging,

[1] https://competitions.codalab.org/competitions/21691

[2] https://montrealartificialintelligence.com/aidebate/

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 645–652
Barcelona, Spain (Online), December 12, 2020.

reaching F1 score 0.673 by combining my own code with two leading open source AI projects. I have released all my code[3] as open source and my experiments can be easily reproduced by anyone with an internet browser.

2 Background

NLP systems are typically built on a pipeline of different pre-processing modules, such as tokenization, sentence segmentation as well as syntactic and semantic analysis. Such pipelines are prone to compounding errors. One solution to these problems is to have fewer couplings. DL allows construction of the end-to-end pipelines reducing language to n-grams or bag-of-words thus replacing compositional semantics with distributional semantics (Boleda, 2019). However, this often results in "linguistically ignorant" models which severely limit the range of tasks that can be performed by the pipeline (Blunsom et al., 2017).

Another approach is to combine multiple modules each specialized in precise execution of one or few NLP tasks, as opposed to the steps in a pipeline. Training of such modules can be done separately or jointly using the same or different datasets (Pruksachatkun et al., 2020). Moreover, the modules can be implemented using different methods of machine learning as appropriate for a particular task. Pretrained DL models can be first fine-tuned on an intermediate task, before fine-tuning again on the target task of interest. These models, also referred to as STILTs can target a narrowly defined model behavior or linguistic phenomenon (Phang et al., 2018).

Linguists distinguish between two different complex constructions involving *if*, tense, aspect, and modality: simple past conditionals vs. past perfect *would*-conditionals. The subtasks 1 and 2 target two substantially different linguistic phenomena, namely counterfactuals:

> -events that did not actually happen or cannot happen

and conditionals (antecedent and consequent):

> -the possible consequence if the events have had happened

However, there is no consensus even among linguists, on treatment of the combined *counterfactual conditionals*. There is no lexical ambiguity of *if*-. There are just different effects produced by the elements with which *if* interacts. It may even be that there is no accurate binary semantic distinction between types of conditionals (Declerck and Reed, 2001) Therefore, the algorithmization of the Task 5 demands an extra-linguistic objective which is clear from its description:

> A counter-factual statement can be converted to a contrapositive with a true antecedent and consequent. Consider the "post-traumatic stress" example discussed above; it can be transposed into "because her post-traumatic stress was not avoided, (we know) a combination of paroxetine and exposure therapy was not prescribed". Such knowledge can be not only used for analyzing the specific statement but also be accumulated across corpora to develop domain causal knowledge (McKinsey and Goodman, 1947)

This objective requires an epistemic modality - *we know*. That is - counterfactual past perfect *would* conditionals (aka "subjunctives") carry additional information as compared to regular past conditionals (aka "indicatives") - the contrapositive conversion merely makes explicit the information already present in its surface form by converting it into its non-modal logical form. Therefore, analyzing the surface form should allow that signal to be detected automatically. I will show how syntactic dependency parsing can be used to that end.

2.1 Related work

Loosely coupled hybrid processing architectures with a clear division between symbolic parsing, connectionist semantic analysis, and symbolic restructuring were considered for a long time as means to overcome the gap between neural and symbolic natural language representations (Wermter, 1997).

[3] https://github.com/lenyabloko/SemEval2020

To enable automatic extraction of the causal relations from text, a new form of shallow semantic parsing called *surface construction labeling* has been recently proposed (Dunietz et al., 2017), which detects not only words and lexical units, but also instances of linguistic *constructions* such as cause-effect pairs. Rather than specifying by hand the constraints and properties that characterize each construction, the proposed DeepCx system relies on ML.

I took a hybrid approach very similar to (Sorgente et al., 2018) which first identifies a set of plausible cause-effect pairs through a set of logical rules based on lexical and structural patterns then it uses Bayesian inference to reduce the number of pairs produced by ambiguous patterns. The SemEval-2010 task 8 dataset challenge has been used to evaluate that model. Unlike that system, ETHAN relies on hand-crafted rules for detecting the counterfactual conditionals.

3 System overview

ETHAN is implemented as "cloud native node" that is essentially a pure algorithm executed in the computing cloud and using tabular data as input and output. Its logic closely follows the aforementioned linguistic interactions between *if* and *would*. According to (Declerck and Reed, 2006), there are nine importantly distinct tense combinations in counterfactuals and its logical forms can be reduced to *canonical* conterfactuals form *if P, (then) Q*, where *P* antecedent and *Q* is its consequent. I further reduced it to combinatorial interactions between the part-of-speech(POS) tags obtained from the sentence constituency parsing (Joshi et al., 2018).

3.1 Experimental setup

The organizers of Task 5 have made available 25000 separate open domain sentences (occasionally pairs) which included transcribed speech with a lot of syntactic and semantic noise, meaning incomplete and ill-formed sentences.

The Subtask 1 includes 13000 sentences manually classified by at least 3 humans as containing (or not) counterfactual statements. This subtask requires a system to then perform classification of the remaining 7000 sentences.

The substask 2 includes 3552 training sentences annotated by humans who identified the causal antecedent and the consequent (if any) present in that sentence by indicating the specific start and end character positions. It then requires the system to automatically annotate the remaining 1950 sentences.

3.2 Subtask 1

One of the latest advancements of DL applications in NLP is so called "transformers" (Wolf et al., 2019) which make possible classification of symbolic sequences (made of POS) based on features that it learns from the large corpus of training data. Although these features are not engineered, it must be linguistically informed. That information or learning signal comes from a combination of patterns implicit in symbolic sequences and annotations made by human experts. In other words, transformers perform *semi-supervised* machine learning (Zhu and Goldberg, 2009). For subtask 1 ETHAN relies on HuggingFace Transformers [4] and pre-trained RoBERTa model (Liu et al., 2019) finetuned on the train dataset for binary classification.

3.3 Subtask 2

For subtask 2 ETHAN relies on Allen Institute for Artificial Intelligence (AI2) constituency parser [5] based on ELMo embeddings (Peters et al., 2018; Joshi et al., 2018). This allows ETHAN to intercept linguistic patterns of interactions between *if* and *would* as discussed earlier. It then puts corresponding textual chunks into sets of candidates for the binary classification task. Note how in this case the hybrid AI becomes braided (neural precedes symbolic which in turn is succeeded by neural filtering). The classification is done by the model finetuned on subtask 1 - just as it is done by the aforementioned STILTs. This "-intermediate" model, however, is not the one that scored highest on the subtask 1.

[4] https://huggingface.co/roberta-base

[5] https://demo.allennlp.org/constituency-parsing

ETHAN's pipeline, however, must continues because the results of classification are not conclusive - there are often more candidate chunks of text than one antecedent and one consequent per sentence. Sometimes all the candidates are classified as counterfactuals or as not counterfactual. However, the filtering out of good candidates causal pairs works remarkably well. This agrees with the results obtained by experiments with STILTs showing that tasks which involve commonsense reasoning are generally useful as intermediate tasks.

Finally, at the end, ETHAN once more applies symbolic logic in order to determine which of the candidate chunks should be classified as antecedents and which as consequents. This time the hybrid AI becomes braided again since ETHAN applies neural attention-based (Dozat and Manning, 2017) dependency parser [6] to each of the candidates in order to determine the direction of causality by following the dependency tree.

Briefly, universal dependency(UD) framework proposes a "universal" annotation scheme that should be applicable to all languages. To date the scheme has served as guidance for the creation of treebanks for more than 70 languages [7]. The annotation scheme is based on an evolution of (universal) Stanford dependencies (de Marneffe and Manning, 2008), Google universal part-of-speech tags (Petrov et al., 2012), and the Interset interlingua for morphosyntactic tagsets (Zeman, 2008). Its main feature is the rule-based transformability into functional head schemes and back which implies that this transformation could be picked up implicitly by a sufficiently complex learning algorithm. However, its potential has not being fully explored (Hernán et al., 2019).

One of the main obstacles is the high cost of annotating UD tree. Considering that it is much cheaper to annotate POS tags than parse trees, it was even suggested that POS tagging alone may avoid using dependency parsing (Zhang et al., 2020). One of the goals I set for ETHAN was to explore the limitations of DL and to show the crucial role of rule-based dependency analysis for causal inference (Pearl, 2018).

3.4 Causal Inference

Identification of causal information from text data is one of the key challenges in NLP (Nazaruka, 2019) that has not been met despite advancements in DL and other statistical approaches. UD framework provides an avenue for combining neural and symbolic approaches by providing semantically reach set of linguistic types which can be used for annotation, learning and causal inference. Specifically, the later can be accomplished by automatically searching dependency trees for a particular type of path. For example, here is a sentence (#202610) from Task 5 train dataset:

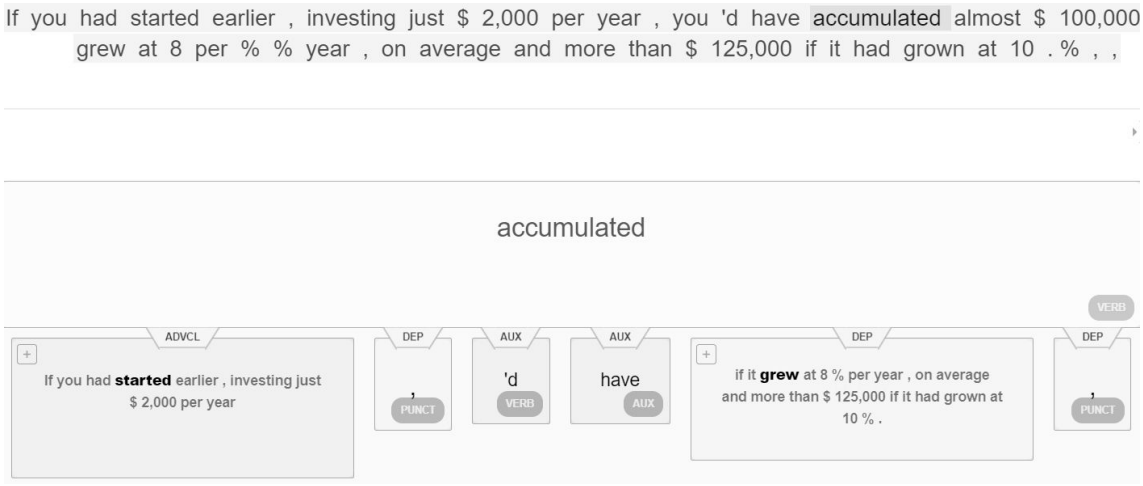

Figure 1: Universal dependency parse

The UD parse of this sentence (Figure 1) is using grammatical types like `PRON`, `VERB`, `ADJ` etc, as well as semantic types like `NSUBJ` which stands for *nominal subject* and `DEP` *unspecified dependency*

[6] `https://demo.allennlp.org/dependency-parsing`
[7] `http://universaldependencies.org/`

(for a complete list of types see Appendix.) The UD scheme is designed to subordinate function words to content words. This allows ETHAN to follow functional dependency between clauses and match the dependants with the candidate chunks of text coming down the pipeline in the output of Subtask 1. So, for example, there are two chunks of text dependent on the verb "accumulated" (bottom left and right of Figure 1). Both chunks contain the *If* but the left one is of `ADVCL` type (that is *adverbial clause modifier*) and contains the verb "started" which opens another branch in the dependency tree. In order to determine which one of the two chunks can be a counterfactual conditional ETHAN follows the dependencies and looks for the `NSUBJ` type clauses - those **not allowed** in the antecedent and thus **point** to the potential consequent. This logic leads to identifying the left side as a potential antecedent. In other words, you can only reach the consequent by going *upwards* the dependency tree and *forward* to the nominal subject.

3.5 Results

For Subtask 1 finetuning the RoBERTa base 12-layer, 768-hidden, 12-heads, 125M parameters model results in F1 score 0.874. Increasing the size of model from base to large 24-layer, 1024-hidden, 16-heads, 355M parameters results in only slightly higher F1 score 0.878. Omitting the details of many experiments I performed to reach this score, the other notable model was XLM [8] for causal language modeling(CLM) in which a given word is trained based only on the previous words and not using the masking technique (Lample and Conneau, 2019).

For Subtask 2 the candidate chunks of text were further classified by XLM-CLM model before passing it to final stage in the pipeline. The resulting F1 score 0.673 reflects the lower precision. Space does not allow me to go into details but the UD-based filtering at the end of the pipeline has identified approximately 1/8 of the proposed causal pairs as a suspect. I have analysed random samples from that set and found few cases where the rules had failed which was expected. However, most of the identified violations were real.

4 Conclusions

DL is very effective at isolated tasks but is not sufficient for causal inference over language. Causal modeling of language requires a modular architecture combining DL models trained on intermediate tasks with rule-based specialist modules. Expansion can be achieved by layering symbolic modules with neural networks on a basis of the universal dependency scheme. Cloud computing provides a flexible and open environment for developing hybrid AI modules at a very low cost.

5 Future work

There exists a large body of work on semantic parsing. Starting with 2014 and 2015 SemEval shared tasks (Oepen et al., 2014), the entire spectrum of approaches has been developed. However, the work is very fragmented and narrowly focused on SOTA efficiency. What's still missing is the unification of semantics in a broader context of AI. Causal reasoning offers a unique opportunity to address the robustness of the approaches against expanding scope and diversity of tasks (Zhang et al., 2019). In my view, the UD scheme provides a context for bringing different ideas together.

Acknowledgements

Special thanks to my daughter for editing this paper!

References

Ernest Wilcox Adams. 1975. *The logic of conditionals: An application of probability to deductive logic*. Number 86. Springer Science & Business Media.

Phil Blunsom, Kyunghyun Cho, Chris Dyer, and Hinrich Schütze. 2017. From characters to understanding natural language (c2nlu): Robust end-to-end deep learning for nlp (dagstuhl seminar 17042). *Dagstuhl Reports*, 7:129–157.

[8] `https://huggingface.co/xlm-clm-ende-1024`

Gemma Boleda. 2019. Distributional semantics and linguistic theory. *ArXiv*, abs/1905.01896.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The stanford typed dependencies representation. In *CF+CDPE@COLING*.

Renaat Declerck and Susan Elizabeth Reed. 2001. Conditionals: A comprehensive empirical analysis.

Renaat Declerck and Susan Reed. 2006. Tense and Time in Counterfactual Conditionals. *Belgian Journal of Linguistics*, 20(January 2006):169–192.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. *ArXiv*, abs/1611.01734.

Jesse Dunietz, Lori S. Levin, and Jaime G. Carbonell. 2017. Automatically tagging constructions of causation and their slot-fillers. *Transactions of the Association for Computational Linguistics*, 5:117–133.

Miguel A. Hernán, John Hsu, and Brian C. Healy. 2019. A second chance to get causal inference right: A classification of data science tasks. *Chance*, 32:42–49.

Vidur Joshi, Matthew E. Peters, and Mark Hopkins. 2018. Extending a parser to distant domains using a few dozen partially annotated examples. *ArXiv*, abs/1805.06556.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *CoRR*, abs/1901.07291.

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. *ArXiv*, abs/1907.11692.

Gary F. Marcus. 2020. The next decade in ai: Four steps towards robust artificial intelligence. *ArXiv*, abs/2002.06177.

J. C. C. McKinsey and Nelson Goodman. 1947. The problem of counterfactual conditionals.

Erika Nazaruka. 2019. Identification of causal dependencies by using natural language processing: A survey. In *ENASE*.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajic, and Zdenka Uresová. 2014. Semeval 2014 task 8: Broad-coverage semantic dependency parsing. In *SemEval@COLING*.

Judea Pearl. 2011. The algorithmization of counterfactuals. *Annals of Mathematics and Artificial Intelligence*, 61:29–39.

Judea Pearl. 2018. Causal and counterfactual inference.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *ArXiv*, abs/1802.05365.

Slav Petrov, Dipanjan Das, and Ryan T. McDonald. 2012. A universal part-of-speech tagset. *ArXiv*, abs/1104.2086.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *ArXiv*, abs/1811.01088.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? *ArXiv*, abs/2005.00628.

Antonio Sorgente, Giuseppe Vettigli, and Francesco Mele. 2018. A hybrid approach for the automatic extraction of causal relations from text.

Stefan Wermter. 1997. Hybrid approaches to neural network-based language processing.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Transformers: State-of-the-art natural language processing.

Xiaoyu Yang, Stephen Obadinma, Huasha Zhao, Qiong Zhang, Stan Matwin, and Xiaodan Zhu. 2020. SemEval-2020 task 5: Counterfactual recognition. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain.

Daniel Zeman. 2008. Reusable tagset conversion using tagset drivers. In *LREC*.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. Broad-coverage semantic parsing as transduction. In *EMNLP/IJCNLP*.

Yifan Zhang, Zhenghua Li, Houquan Zhou, and Min Zhang. 2020. Is pos tagging necessary or even helpful for neural dependency parsing? *ArXiv*, abs/2003.03204.

Xiaojin Zhu and Andrew B. Goldberg. 2009. Introduction to semi-supervised learning. In *Introduction to Semi-Supervised Learning*.

A Appendix

Type	Description
acl	clausal modifier of noun (adjectival clause)
advcl	adverbial clause modifier
advmod	adverbial modifier
amod	adjectival modifier
appos	appositional modifier
aux	auxiliary
case	case marking
cc	coordinating conjunction
ccomp	clausal complement
clf	classifier
compound	compound
conj	conjunct
cop	copula
csubj	clausal subject
dep	unspecified dependency
det	determiner
discourse	discourse element
dislocated	dislocated elements
expl	expletive
fixed	fixed multiword expression
flat	flat multiword expression
goeswith	goes with
iobj	indirect object
list	list
mark	marker
nmod	nominal modifier
nsubj	nominal subject
nummod	numeric modifier
obj	object
obl	oblique nominal
orphan	orphan
parataxis	parataxis
punct	punctuation
reparandum	overridden disfluency
root	root
vocative	vocative
xcomp	open clausal complement

Table 1: Universal Dependency types.