Articles Field research

Supervised learning, in the context of artificial intelligence ([AI](https://searchenterpriseai.techtarget.com/definition/AI-Artificial-Intelligence)) and [machine learning](https://searchenterpriseai.techtarget.com/definition/machine-learning-ML), is a type of system in which both input and desired output data are provided. Input and output data are labelled for classification to provide a learning basis for future data processing. The term supervised learning comes from the idea that an algorithm is learning from a training dataset, which can be thought of as the teacher.

# **Default Reasoning in NLP (Uri Zernik, Allen Brown)**

## Intro

When equipped with comprehensive linguistic knowledge, the system may fare worse than when equipped with impoverished linguistic knowledge, since **additional rules can impair simple cases.** Ex:

1. The child sold his parents’ dearest ornament
2. The child sold to a stranger was alive

Linguistic systems may handle both cases: **a** which is unambiguous, and **b** whose interpretation is obscured by its grammatical construction. Sentence **b** involves the  *garden path* phenomenon, in which the reader is led to a standard interpretation which is retracted towards the end of the sentence. For a computer parser, it’s important to show **how the initial interpretation is flipped in parsing sentence 2**, and **that for sentence a no flipping is required**. This cannot be achieved by prolog-based parsers.

## The theoretical issue

1. *Ambiguity:*

Ex: Mary was tired, so John showed her home.

* Semantics: what John (if there are more?)
* Syntax: what POS is “her”?
* Lexicon: what is “to show” and what’s its meaning?
* Role-binding: does “show” (=make something visible) takes one or 2 objects?
* Context: what’s the purpose of the act?

1. *Non-monotonicity:*

Garden-path sentences highlight the problem. Ex:

* *The old man’s glasses were filled with sherry.*

The semantic assumption “glasses” must be retracted at the end, since they turn out to be drinking containers

* *I saw the Grand Canyon flying to New York.*

Syntactic assumption dictates the default structure (S -> VP, NP), taking the Grand Canyon as the agent flying to NY. Retracted at the end after showing the knowledge conflict, since canyons can’t fly.

* *The book sold yesterday was cheap.*

The word sold is assumed to take active voice. Backtracking occurs at the end due to knowledge of the selling act – books cannot sell days.

* *The horse raced past the barn fell.*

Two assumprions which fail:

* “raced” taken by default as active voice
* “raced past the barn” taken as verb phrase of the sentence.

Both fail when encountering “fell”, and “raced” is taken as past participle in the usual sense of “being driven”.

1. *Context dependency:*

* *John wanted a new car. He took it up with his dad.*
* *This is his new Porsche. He took it up with his dad.*

In the **first** sentence, “up” is part of the “took it up with somebody” lexical phrase. In the **second**, it serves as a general adverb (up = up the hill).

# **Event2Mind: Commonsense Inference on Events, Intents, and Reactions (Hannah Rashkin, Maarten Sap, Noah A. Smith, Yejin Choi – Allen institute for AI)**

## Intro

Understanding a narrative requires commonsense reasoning about the mental states of people in relation to events. For example, if “Alex is dragging his feet at work”, pragmatic implications about Alex’s intent are that “Alex wants to avoid doing things”. Furthermore, while not explicitly mentioned, we can infer that people other than Alex are affected by the situation, and these people are likely to feel “frustrated” or “impatient”.

This approach is used for NLP applications where an accurate anticipation is needed for determining a person’s intentions or emotions without them mentioning explicitly. Ideally, the system should infer and reason with the human’s mental state based on **the events the user has experienced,** without the user saying how it’s feeling.

# **Deep contextualized word representations (Mathew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer) – Allen Institute for Artificial Intelligence, University of Washington**

## Abstract

It represents a deep contextualized word representation which covers two main issues:

* Complex characteristics of word use: semantics & syntax
* Their variation over different linguistic contexts

The word vectors are learned functions of the internal state of a **bidirectional language models** (biLM) and are used for enhancing different challenges in NLP such as question answering, textual entailment, sentiment analysis, etc.

## Introduction

In many neural language understanding models, two challenges must be modeled:

* Complex characteristics of word use: semantics & syntax
* Their variation over different linguistic contexts

What the paper brings as innovation is the upgrade from the traditional word type embeddings, by assigning to each token a representation which is described as being function of the whole input sentence. The representations are called **ELMo**, which stand for Embeddings from Language Models. The efficiency of ELMo representations is proven by their practical use, because they can be easily coupled with the existing models for six challenges in language understanding problems, which include textual entailment, question answering, and sentiment analysis.

## Related work

Pretrained word vectors are a commonly met feature of the NLP architectures, including the three mentioned above. The main reason for their inclusion is the ability of capturing semantic and syntactic meaning for words coming from large text portions that have no labelling. There have been previous attempts in designing this kind of representation, but they implied either enhancing the words with additional information (e.g., Wieting et al., 2016; Bojanowski et al., 2017), or learning separate vector representation for different contexts in which the word may occur.

Other similar tools which are also centered on context-dependent representations, such as **context2vec**, use a **Long Short Term Memory (LSTM)**, to encode the context around a single reference word (pivot).

A key remark of the previous representation implementations show that different layers of a deep biRNN encode different types of information. For example, if multi –task syntactic supervision such as POS tagging is placed at the lower levels of a deep LTSM, the overall performance is improved on the higher level tasks such as dependency parsing (Hashimoto et al., 2017) or CCG super tagging (Søgaard and Goldberg, 2016). Example: in an RNN-based encoder-decoder machine translation system, Belinkov et al. (2017) showed that, in a 2-layer LTSM encoder:

* **First layer**: better at predicting part of speech (POS) tags than the second layer;
* **Top/Second layer:** learns representations of the word sense.

In the case of Dai and Le (2015) and Ramachandran et al. (2017) pairs of the encoder-decoder type are pre-trained using language models and sequence auto encoders and then fine tune with task specific supervision. **As opposed to that,** the proposed solution fixes the weights and adds additional task-specific model capacity after pre-training the biLM with unlabeled data. This allows for leverage large and complex universal biLM representations for cases where downstream training data sizes dictates a smaller supervised model.

## ELMo: Embeddings from Language Models

Unlike most widely used word embeddings, ELMo word representations are calculated as functions of the entire input sequence. They are computed on top of 2-layer biLMs with **character convolutions**, as a linear function of the internal network states. What this setup favours is that it allows us to do semi supervised learning, where biLM is pre-trained at a large scale and easily incorporated in existing neural NLP architectures.

# **Open Information Extraction – Allen Institute of AI**

## Task formation (tuple building)

The proposition is modeled as a **tuple** consisting of a single predicate operating over a non-empty set of arguments. The predicate and arguments are **continuous spans** from the sentence. As in traditional Open IE, every tuple should be asserted by the sentence and the order of the elements should be such that it would be naturally interpretable when reading form left to the right. This kind of approach leads to BIO tagging: the set of Open IE tuples for the sentence S are grouped by predicates p of type head-word.

Advantage: grouping the tuples this way allows to run the model once for every predicate head, and accumulate the predictions across predicates to produce the final set of extractions. **Compared to SRL**, open IE uses multi-word predicates that often incorporate modals and embedded predicates. **Multiple extractions per predicate** are encoded by assigning the same argument index to all arguments appearing in that position across all of the predicate’s extractions. That’s why on different examples there are the same argument notations.

## Inference

At inference time, we first identify all verbs and nominal predicates in the sentence as **candidate predicate heads**. We use a Part Of Speech (POS) tagger to **identify verbs**, and Catvar’s subcategorization frames (Habash and Dorr, 2003) for **nominalizations**, identifying nouns which share the same frame with a verbal equivalent (e.g., acquisition with acquire). We then generate an input instance for each candidate predicate head. For each instance, we tag each word with its most likely BIO label under the model, and reconstruct Open IE tuples from the resulting sequence according to the method described in Section 3, with the exception that we ignore malformed spans (i.e., if an A0-I label is not preceded by A0-I or A0-B, we treat it as O).