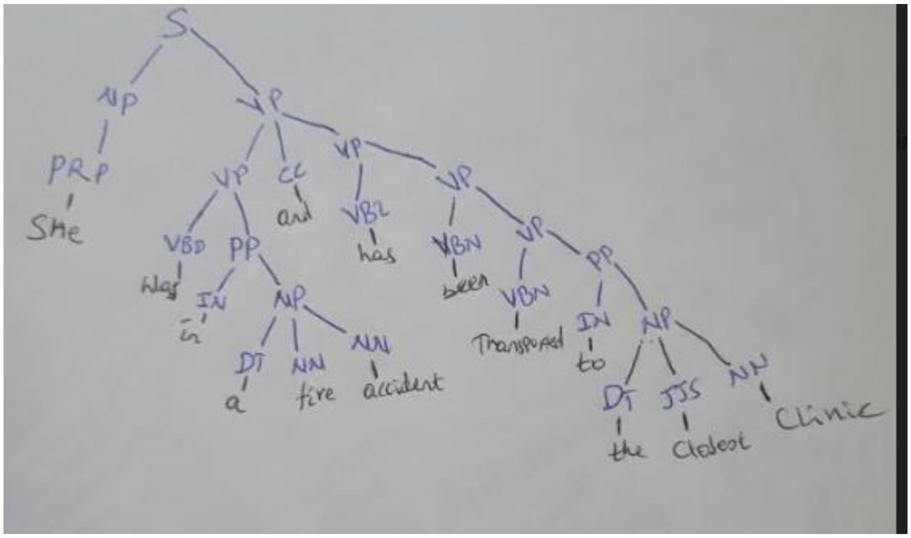
Sentence: She was in a fire accident and has been transported to the closest Clinic.

PSG TREE



Phase Terms and their Definitions:

CC: Coordinating Conjunction

DT: Determiner

IN: Preposition

JJS: Adjective, Superlative

NN: Noun, Singular

NP: Noun Phrase

PRP: Personal Pronoun

PP: Prepositional Phrase

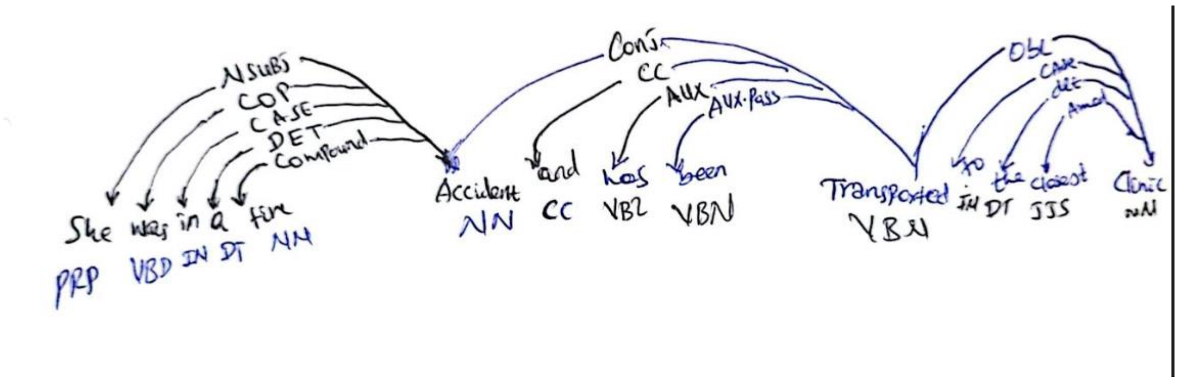
VBD: Verb, Past tense

VBN: Verb, past participle

VBZ: Verb, third person singular present

VP: Verb Phrase

Dependency Parse:



Phase terms and Their Definitions

NSUBJ: Nominal Subject

COP: Copula

COMPOUND: Compound

CONJ: Conjunct

CC: Coordination

AUX: Auxiliary

AUX.PASS: Passive Auxiliary

OBL: Indirect Nominal

CASE: Case Marking

DET: Determiner

AMOD: Adjectival Modifier

SRL PARSE:

Verb 1: was

Predicate: was in a fire accident

ARG0: She

ARG1: in a fire accident

The relationship of the arguments is that ARG0 is the agent of the sentence and ARG1 is the thing that happened on this ARG0.

Verb 2: Transported

Predicate: Transported to the closest clinic.

ARG0: She

ARGM: to the closest clinic.

Tag of the modifier: GOL

GOL = Final destination motion verbs

The relation between the argument and modifier are:- Arg0 is the agent of the sentence and

ARGM is where the agent ended up.

Pros and cons of Parsers

Compared to the other two parsers, I believe PSG tree was simpler to use and produced a list of speech parts that many machine learning algorithms can use; However, PSG tree does not look at the sentence's grammatical structure. This gave us a list of POS and explored the dependencies between the words, which is better than the PSG tree and is more useful for applications that require grammatical analysis. For the dependency parsing, it was difficult to determine the dependency relations between each word. Finally, although SRL parsing is simple to implement, I am unsure whether it is as useful for machine learning algorithms as it is for identifying various actors and aspects of an action but not the role of each word.