CONTEXT IDENTIFICATION

A project report submitted to College of Engineering Trivandrum
in partial fulfilment of degree of

B.Tech.
in
Computer Science and Engineering

Submitted by

Anashka Naina
Mohammed Shiraz A K
Rohit P
Sarath Sreedhar

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING TRIVANDRUM

KERALA

Abstract

Since the 20th century computer have taken over the control of all domain. The proof of this is the widespread usage of Deep Learning and Natural Language. Deep Learning models learn features from training data and use this models to perform specific tasks. Natural Language Processing concerns with the interaction between human beings and computer.

The context is the words that surround other words and impact their meaning or setting in which something occurs.The three pillars holding NLP is representation, context and reasoning. For any action, the reasoning based on the knowledge and contextual information produce better results. Current methods of NLP are largely driven by classification and predictive models.

The importance of context based search is categorizing documents based on domain.The use of deep learning models in this particular domain helps in getting accurate results.For any application like word sense disambiguation,word prediction , context understanding is a vital ingredient.

Here we describes an approach of context identification with the help of a global knowledge base which have better representation and reasoning ability.

Contents

Chapter 1

Introduction

A context model (or context modeling) defines how context data are structured and maintained (It plays a key role in supporting efficient context management). It aims to produce a formal or semi-formal description of the context information that is present in a context-aware system. In other words, the context is the surrounding element for the system, and a model provides the mathematical interface and a behavioral description of the surrounding environment.
Contextual awareness means knowledge of the situation. We know that Reasoning based on the knowledge and contextual information produce better results for any action. When we are processing a Natural language text, it needs to be processed in right context.Current methods of NLP are largely driven by computational statistics.These methods don't attempt to understand the text.Instead convert the text into data and then attempt to learn from the pattern in that data.

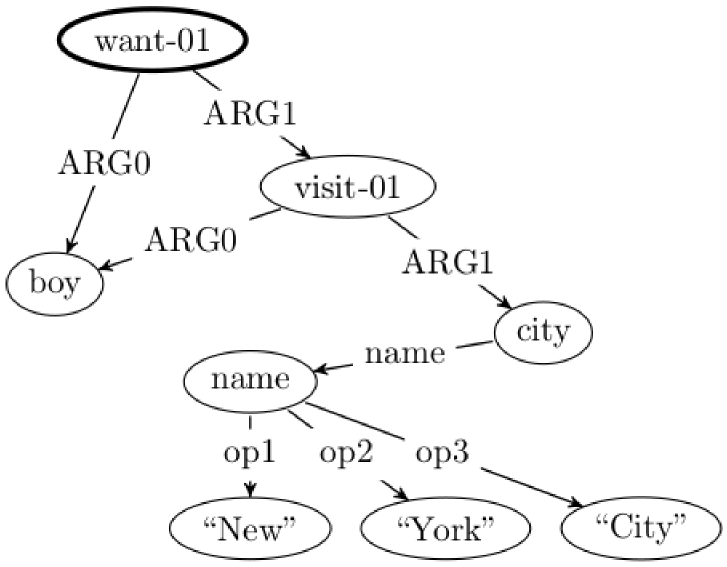
Chapter 2

Literature Survey

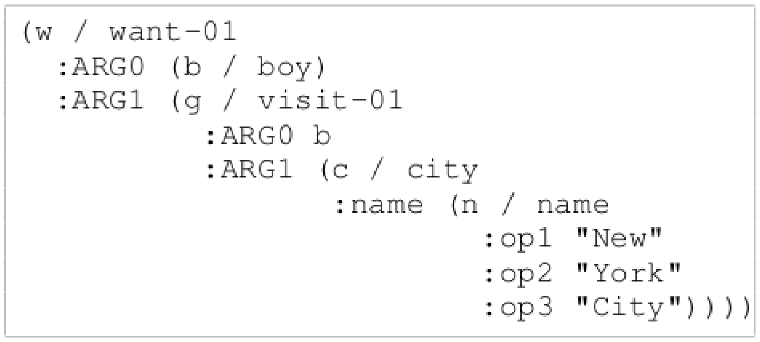
A Discriminative Graph-Based Parser for the AMR

AMR Parser focuses on mapping natural language strings into meaning representation. Abstract Meaning Representation (AMR) (Banarescu et al., 2013(**?**)) is a graphical representation of sentence meaning. Identifying the meaning with the help of a parser include two stage processing of input sentence. First stage is to identify the concept and we can use nodes in the graph to represent the same. In the second stage we need to relate the concept nodes created in the first stage based on the relation acquired from sentence. Edges in the graph can be used for representing the relations.

JAMR is the first published system for automatic AMR Parsing. The name discriminative came because the system is based on a statistical model whose parameters are trained discriminatively using annotated sentences in the AMR Bank corpus. The core of JAMR is a two-part algorithm that first identifies concepts AND relatiion between them. A semi-Markov model is used for concept identification. The relations is identified by searching for the maximum spanning connected subgraph (MSCG) from an edge-labeled, directed source graph. The source graph representing union of all the possible relations between the identified concepts.



(a) Graph.



```
(w / want-01
  :ARG0 (b / boy)
  :ARG1 (g / visit-01
          :ARG0 b
          :ARG1 (c / city
                 :name (n / name
                         :op1 "New"
                         :op2 "York"
                         :op3 "City"))))
```

Two equivalent ways of representing the AMR parse for the sentence, ”The boy wants to visit New York City.”

Notation And Overview

AMR parsing represents an AMR parse as a graph $G =< V, E >$ vertices and edges are given labels from sets L_V and L_E , respectively. G, the graph constructed involves two stages. The first stage identifies the concepts evoked by words and phrases in an input sentence $w =< w_1, \cdots, w_n >$, each w_i a member of vocabulary W . The second stage connects the concepts by adding L_E -labeled edges capturing the relations between concepts, and selects a root in G corresponding to the focus of the sentence w.

Concept identification involves segmenting w into contiguous spans and assigning to each span a graph fragment corresponding to a concept from a concept set denoted by F. We use a sequence labeling algorithm to identify concepts.

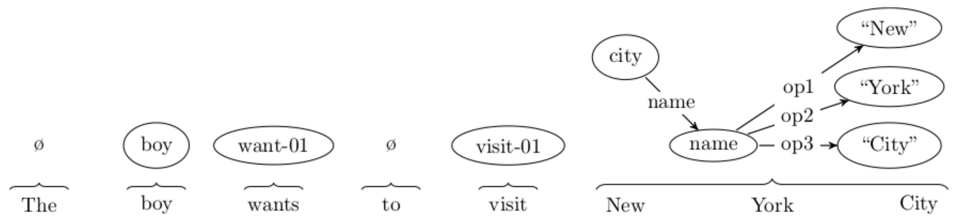
The relation identification stage is similar to a graph-based dependency parser. In dependency parser we finding the maximum-scoring tree over words. Instead of that relation identification stage finds the maximum-scoring connected subgraph that preserves concept fragments from the first stage. The link between pair of vertices is at most one edge, and is deterministic with respect to a special set of edge labels $L*_E{\subset}\mathrm{L}_E$. The set $L*_E$ consists of the labels ARG0-ARG5, and does not include labels such as MOD or MANNER.

Concept Identification

The concept identification stage maps spans of words in the input sentence w to concept graph fragments from F , or to the empty graph fragment ϕ.

These graph fragments often consist of just one labeled concept node, but in some cases they are larger graphs with multiple nodes and edges. The label are from Propbank frame-set.

Concept identification is illustrated in Figure 2 using sentence, "The boy wants to visit New York City."

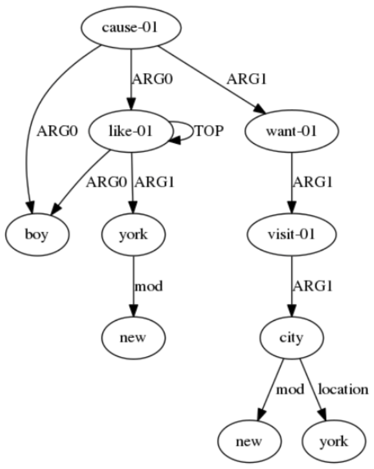


Concept labeling for sentence "The boy wants to visit New York City."

In the concept identification stage each span of words identified as concept with the help of sequence labeling algorithm is mapped with candidate graph fragment created at the training phase of the parser. The mapping is given by the function $clex : W^* \rightarrow 2^F$, which provides candidate graph fragments for sequences of words. Formally, the concept identification is done through following steps:

- The segmentation of w into contiguous spans represented by boundaries b, giving spans $< w_{b0:b1}, w_{b1:b2}, \cdot\cdot\cdot w_{bk-1:bk} >$, with $b_0 = 0$ and $b_k = n$.

- An assignment of each phrase $w_{bi-1:bi}$ to a concept graph fragment $c_i \in clex\ (w_{bi-1:bi}) \cup \{\phi\}$.



Need of score calculation explained using sentence "The boy likes New York so he wants to visit New York City."

It is possible that multiple candidate graph can be mapped to same span of words even though its not a full match. Consider figure 2, here there is two name entity is present, "New York" and "New York City". Without score calculation both named entity will map to same graph fragment.

The equation to calculate the scores of each mapping between word sequence and concept graph fragments is :

$$score(b, c; \theta) = \sum_{i=1}^{k} \theta^T f(w_{b_{i-1}:b_i}, b_{i-1}, b_i, c_i)$$

Following are the features considered in the function $f(w_{b_{i-1}:b_i}, b_{i-1}, b_i, c_i)$:

- **Fragment given words** : Probability of a concept graph fragment given the sequence of words in the span.

- **Length** of the matching span.
- **NER** : 1 if the named entity tagger marked the span as an entity, 0 otherwise.
- **Bias** : 1 for any concept graph fragment from F and 0 for $\emptyset$.

From all the mapping between b and c, the highest-scoring b and c is found using a dynamic programming algorithm: the zeroth-order case of inference under a semi- Markov model. Let $S(i)$ denote the score of the best labeling of the first i words of the sentence, $w_{0:i}$; it can be calculated using the recurrence:

$$S(i) = 0$$

$$S(i) = \underset{\substack{j:0 \leq j < i, c \epsilon \\ clex(w_{j:i}) \cup \phi}}{max} \left\{ S(j) + \theta^T f(w_{j:i}, j, i, c) \right\}$$

Relation Identification

The relation identification stage adds edges among the concept subgraph fragments identified in the concept identification stage. The idea is to crate a multigraph by taking union of all the subgraph fragments identified in the concept identification stage. Then using Maximum Preserving, Simple, Spanning, Connected Subgraph Algorithm to find best relationships which results in AMR of the input sentence.

The fully dense labeled multigraph D that includes the union of all labeled vertices and labeled edges in the concept graph fragments, as well as every possible labeled edge $u \rightarrow v$, for all u, $v \in V_D$ and every $l \in L_E$, is created. From the created multigraph D a subgraph $G =< V_G, E_G >$ that respects the following constraints is generated with the help of MSCG Algorithm:

- **Preserving** : All graph fragments from the concept identification phase are subgraphs of G.
- **Simple** : At most one edge between any two nodes.
- **Connected** : Every edge is accessible from every other vertex.
- **Deterministic** : Edge with a particular label must be once.

Given the constraints, the maximum- scoring subgraph is obtained. The definition of score to decompose by edges, and with a linear parameterization is given by:

$$score(E_G; \psi) = \sum_{e \epsilon E_G} \psi^T g(e)$$

MSCG Algorithm

The steps for constructing a maximum preserving, simple, spanning, connected (but not necessarily deterministic) subgraph is as follows. These steps ensure that the resulting graph G satisfies the constraints: the initialization step ensures that the preserving constraint is satisfied, the preprocessing step ensures that the graph is simple, and the core algorithm ensures that the graph is connected.

- **Initialization** : Let $E(0)$ be the union of the concept graph fragments weighted, labeled, directed edges. Let V denote its set of vertices. Let V denote its set of vertices.

- **Pre-processing** : We form the edge set E by including just one edge from $E(D)$ between each pair of nodes. For any two nodes u and v, include only the highest scoring edge between u and v.

- **Core Algorithm** : Maximum Preserving, Simple, Spanning, Connected Subgraph (MSCG) Algorithm.

MSCG Algorithm

[1] MSCGlet $E^1 = E^0 \cup \{$ e $\in$ E $\mid \psi^T$ g(e)>0 $\}$; create a priority queue Q containing remaining edge prioritized by score; i = 1; Q nonempty and (V, $E^{(i)}$) is not yet spanning and connected $i = i + 1$; $E^{(i)} = E^{(i-1)}$ e = arg $max_{e \in Q}$ ψ^Tg($e^{'}$); remove e from Q; if e connects two previously unconnected components of (V, $E^{(i)}$) then add e to $E^{(i)}$ **return** G

Abstract Meaning Representation for Multi-Document Summarization

Method proposed in this paper aims to generate abstract summarization of multiple document with the help of Abstract Meaning Representation Graph. AMR is a meaning representation formalism which use nutrelized words of the sentences as concept. AMR also draw the relationship between the identified concepts to form a single rooted, directed, acyclic graph.

In traditional abstract summarization the idea is to generate ”semantic units” from the input document and reconstruct sentences of them. Previous works like noun/verb phrase, word-occurrence graph, syntactic parse tree and domain specific-templates have difficulty in reconstruction of the meaningful sentences from the building blocks.

The proposed architecture has three major components : source sentence selection, content planning and surface realization.

- **Source sentence selection** : takes a set of news articles as input and selects sets of similar sentences covering different aspects of the topic.
- **Content planning** : consumes a set of similar sentences and derives a summary graph from them.
- **Surface realization** : transforms a summary graph to a natural language sentence.

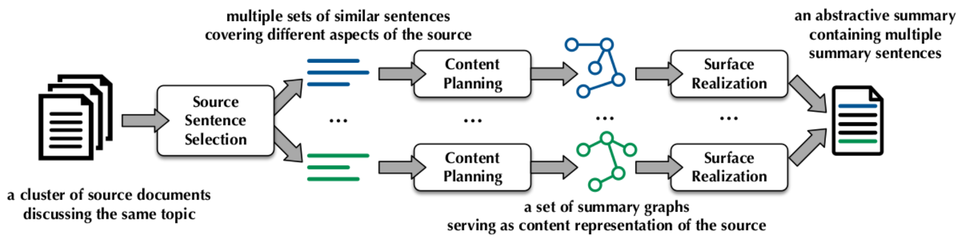

multiple sets of similar sentences covering different aspects of the source
an abstractive summary containing multiple summary sentences
Source Sentence Selection
Content Planning
Content Planning
Surface Realization
Surface Realization
a cluster of source documents discussing the same topic
a set of summary graphs serving as content representation of the source

Figure 2.1: System framework, consisting of three major components

Approach

Content planning take set of similar sentences as input, then AMR graph is created for each sentence. The AMR graphs are then merged into a connected source graph and from the source graph with a structured prediction technique the summary graph is outputted. Finally the surface realization step convert this summary graph into its PENMAN representation to generate summarized natural sentences of the multi document.

Content Planning

The source sentence is converted into AMR graph. The graph construction is done with the help of an AMR Parser. JAMR (Flanigan et al., 2014) and CAMR (Wang et al., 2015b) are two parsers that can be used to measure the impact of AMR parsers in Abstract Summarization.

JAMR

JAMR is the first open-source AMR parser which generate the AMR graph in two steps. First stage is the concept identification and the second stage is relationship identification. Each stage is a discriminatively-trained linear structured predictor with rich features that make use of part-of- speech tagging, named entity tagging, and dependency parsing.

To train the parser, spans of words must be labeled with the concept fragments they evoke. The concept or span of words from the input sentence is generated with the help of sequence labeling algorithm like Markov model. Then a mapping between concept graph fragment and the span of words is obtained based on feature rich scoring mechanism.

The second stage in JAMR is to identify the AMR graph through relationship identification. In this stage Maximum Preserving, Simple, Spanning, Connected Subgraph Algorithm use used to generate the AMR graph from a source graph which is the union of all the concept graph fragments.

CAMR

It is a two-stage framework for AMR Parsing. First stage involves generation of a dependency tree of the input sentence with the help of a dependency parser. And in the second stage a transition based algorithm will convert the dependency tree to an AMR graph.

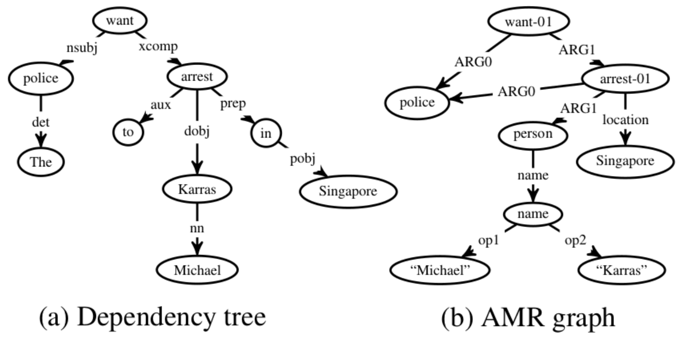


Figure 2.2: Dependency tree and AMR graph for the sentence, "The police want to arrest Micheal Karras in Singapore."

CAMR uses three type of action for the transition phase. With the help of these type of action the dependency tree is converted in to AMR graph. Figure 2.2 shows the example of generated AMR Graph from the dependency tree.

- Actions performed when an edge is visited.
- Actions performed when a node is visited.
- Actions used to infer abstract concepts in AMR that does not correspond to any word or word sequence in the sentence.

Source Graph Construction

Given a set of source sentences and their AMR graphs, source graph construction attempts to consolidate all sentence AMR graphs to a connected source graph through a process named concept merging. The merging is performed on common concept nodes present in sentence AMRs. Coreference resolution helps in identifying clusters of mentions of same entity. Finally, a 'ROOT' node is introduced; it is connected to the root of each sentence AMR graph, yielding a connected source graph.

Summary Graph Extraction

A summary graph which containing the salient content of source texts, is identified from the source graph with the help of a trainable, feature-rich structured prediction framework. Two iterative function performed by the framework are graph decoding and parameter update.

- **Graph Decoding** : Identifies an optimal summary graph using integer linear programming.
- **Parameter update** : Performs parameter update by minimizing a loss function that measures the difference between the system-decoded

summary graph and the gold standard summary graph.

The source graph is represented by $G = (V, E)$. Two binary variable v_i and $e_{i,j}$ indicate whether node v_i and edge $e_{i,j}$ are selected or not. $f(i)$ and $g(i, j)$ represent the set of features that describes the importance of node and edge respectively. θ and ϕ are feature weights. The scoring function for graph G is given by following equation.

$$score(G; \theta, \phi) = \sum_{i=1}^{N} v_i [\theta^T f(i)]_{v_i} + \sum_{(i,j) \in E} e_{i,j} [\phi^T g(i, j)]_{e_{i,j}}$$

The features considered to calculate score of source graph using above equation include :

- Concept/relation labels and their frequencies in the documents.
- Average depth of the concept in sentence AMR graphs.
- Whether the concept is a named entity/date entity.
- The average length of concept word spans.

Surface Realization

The surface realization component converts each summary graph to a natural language sentence. This is a significant stage because AMR do not accommodate syntactic structure in the graph construction. First AMR is converted to PENMAN format and then using AMR-to-text generator English sentences are formed.

Chapter 3

Project Methodologies, Results and Achievements

3.1 Methodology

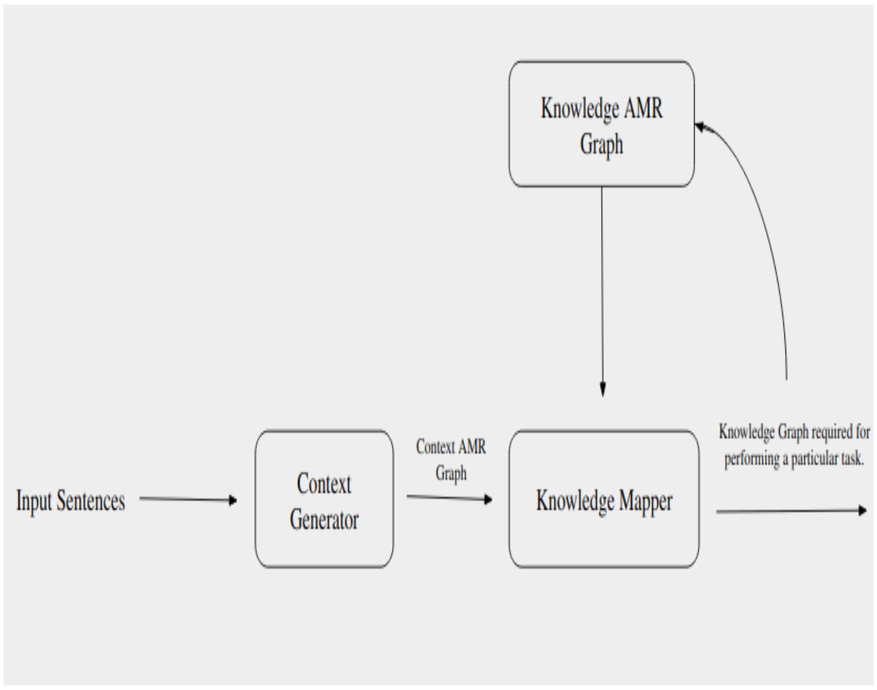
We consider a standard parser model, which takes in input X, to form a meaningful representation ¡Ci, Ri, Cj¿, and then form an understanding of the representation by incorporating the existing knowledge base B (created by initial learning procedures) and to generate the global context Y of input X as our target value of our parser. Understanding the representation involves context retrieval from B and linking the extracted representation ¡Ci, Ri, Cj¿ with context Y through native graph storage.

Storage and retrieval of context Y from knowledge B is the main problem that we face in the implementation of the knowledge base B. Native graph storage make use of the Index-free adjacency property of native graph database helps in fast storage and retrieval.

Given an input, the parser would first analyze and generate the most appropriate representation of the input sentence and this representation is mapped to the native graph from and linked with the knowledge base B. The representation that we are using is the Abstract Meaning Representation(AMR) which is DAG representation which is rooted, labelled, directed and acyclic. The issue with this representation is that it is biased towards the English language and the scope for multi-linguality is limited.

The model of the parser is designed towards making use of the context from the knowledge base B, and use it to understand and reason appropriately. The model starts by taking the input sentence or paragraph and converting it to the appropriate representation and using the existing knowledge base to reason and generate the required result.

3.2 Model



Input Sentences

Input to the model is a textual document preferably in PDFformat. The text is extracted from PDF by using existing tools like PyPDF2.Initially the word limit may be restricted to 1000 words per input

Context Generator

Context Generator generates context from textual data using a deep learning model capable of extracting contextual features from it.An meaningful representation of context is necessity for faster retrieval of contextual information from texts.

Context and Knowledge AMR graph

AMR is useful in representing sentence which removes syntactic idiosyncrasies.AMR graph based sentences can be easily summarized.The representation used in knowledge base and output from context generator are AMR graph.Existing parser like JAMR or CAMR is used to convert sentence into AMR.

Knowledge Mapper

Knowledge Mapper maps context with their reasoning.Context is substantiated by situation information derived from text.Knowledge base is a self evolving information database

Output

The output is text file which provides values for contextual feature because complex representation makes it hard for end user to understand

3.3 Datasets

- News headlines published over a period of 15 years. - Kaggle Challenge
- AynRand Novel "The fountainhead"

3.4 News headline dataset

Format: CSV ; Single File

- publish_date is Date of publishing for the article in yyyyMMdd format

- headline_text is Text of the headline in Ascii , English , lowercase

Start Date: 2003-02-19 End Date: 2017-12-31

Total Records: 1,103,663

3.5 Accuracy

The headlines are extracted from several GB of raw HTML files using Jsoup, Java and Bash. The entire process takes 11 minutes.

This logic also : chooses the best worded headline for each article (longest one is usually picked) ; clusters about 17k categories to 200 large groups ; removes records where the date is ambiguous (9k cases) ; finally cleans the selected headline via a string 'domestication' function (which I use for any wild text from the internet).

The final categories are as per the latest sitemap. Around 1.5k rare categories remain and these records (20k) can be filtered out easily during analysis. The category is unknown for 200k records.

Similar news datasets exploring other attributes, countries and topics can be seen on my profile.

3.6 The Fountainhead

It consist of 59 chapters. A chapter provides around 200 sentences. The dataset contains raw textual file in english. It can be used for unsupervised and rule based learning activities.

Chapter 4

Conclusion

The context is the circumstances that form the setting for an event, statement, or idea, and in terms of which it can be fully understood. Our Aim was to develop a never ending learner which improves its context identification ability.

Grammatical choices in speech and writing are made in response to the opportunities and constraints provided by the context in which they occur, and in turn contribute to context. By observing grammatical variations in different contexts we can learn more about those contexts and, conversely, by studying relevant features of context we can learn about their influence on grammar.

Bibliography

[1] Ann Hewings and Martin Hewings . Grammar and Context, An Advanced Resource Book , 2005.

[2] Daniel Jurafsky and James H. Martin . Speech and Language Processing,An Introduction to Natural Processing,Computational Linguistics,and Speech Recognition, 2008.

[3] `https://amr.isi.edu/a.pdf`