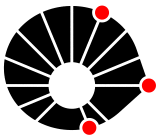

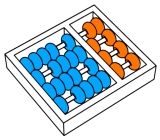

Semantic Metadata Extraction from Subtitles of Video Lectures

Guilherme P. Gribeler, Julio Cesar dos Reis

Relatório Técnico - IC-PFG-18-25
Projeto Final de Graduação
2018 - Dezembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Semantic Metadata Extraction from Subtitles of Video Lectures

Guilherme Gribeler, Julio Cesar dos Reis*

December 2018

Abstract

Video lectures can stimulate learning experiences considering individual needs and learning styles. Extracting relevant information from video lectures can be useful to recommendation purposes and to interpret a concept in a exact moment of a lecture that a student can be interested in watching. The extraction of semantic metadata from a video natural language subtitle involves challenges in dealing with informal aspects of language and the detection of semantic classes from the free text. In this work, we propose a technique for extracting semantic metadata, which consists in developing a tool to extract the subtitle of a *YouTube* video in a text file and then use semantic annotation tools to identify semantic classes from the subtitle text file. We conduct an evaluation to compare the effectiveness of distinct semantic annotation tools on this task. Obtained results indicate that both AutoMêta tool and the our proposed SubAnnotator tool can perform the task of semantic annotating relevant terms well, but Ontotext and NCBO are not very effective for accomplishing this task. The difference between SubAnnotator and AutoMêta is the ability of annotating multiple occurrence of the terms throughout the input text. The SubAnnotator was able to annotate a higher number of occurrences than AutoMêta. The results also indicate that the biggest challenge on the video lectures semantic metada extraction process is the definition of the ontology used by the tools.

1 Introduction

Technology has drastically changed the way we live. It has a huge impact on many aspects of our lives. For instance, we can send a message to someone in a matter of milliseconds. If we were not able to communicate so fast we definitely would be living in a totally different society. In this context, the use of technology to turn available education content is irreversible. With the growth of information dissemination, the Web has played an important role for accessing multimedia content that helps in the learning process.

Such information exchange helps us to come up with solutions to hard problems making our lives easier. Another important aspect to be considered to come up with these solutions is the way we pass our knowledge to other people. Technology can help to spread knowledge

*Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP

in many different ways. One of them is through video lectures. Any teacher who has access to a camera can record his/her lecture and post it online if he/she has internet access. Anyone with internet access can watch the recorded lecture as a student and learn something new.

Nowadays, the number of lectures available online has grown very fast, which turns difficult for someone who wants to learn something new to chose which is the best video to watch. Indeed, huge efforts are still required by people to select the appropriate videos in the learning process and to identify those similar videos covering the same or related concepts. It would be useful if we could recommend the best video lecture to people based on a few keywords which describe what the person wants to learn.

There are several issues associated to this task because handling natural language aspects from video lecture subtitles remains very difficult. Usually, automatic techniques are limited to only process the lexical syntactical aspects of the free texts. This prevents understanding the meanings associated to the used terms in the video lecture. Extracting metadata from video lectures is a complicated task because each lecture is different. Each lecture has a topic which can be broader or more specific; and the length of the lectures are different. Each teacher has its on way on presenting the lecture which can be more or less formal. All of these aspects influence the number of words that are significant to the problem of describing what the lecture is about as good as possible.

In this context, analyzing the video's audio is a hard task. A possible approach is to convert the video's audio in text and run the analysis on the text instead of the audio itself. This conversion is not an easy task. A few tools can be used to achieve this goal such as the YouTube Captions API [5], oTranscribe [3] and Trint [4]. These tools vary in the language and the audio's file size they accept and in the quality of the resulting text. Having the text, the automatic identification of the metadata is still an open research challenge. The issues concern handling the information free text and the language aspects. For example, the Portuguese language is hardly studied. Although some semantic annotation tools such as AutoMêta [1], OntoText [13] and NCBO [10], can be used to extract this data, the quality of the resulting metadata varies from tool to tool. In addition, there is no way of determining which tools are the best to complete a specific task.

This investigation aims to develop a sofware tool which enables to extract relevant semantic metadata from video lectures. These metadata must be able to describe the video well so it could be used as input to some recommendation tool. Our developed software tool takes a *URL* of a video lecture from *YouTube*, automatically extract the relevant semantic metadata from it and then link them to the exactly moment when the semantic metadata is cited on the video. The semantic metadata refer to semantic annotations in natural language texts from the video subtitle content. The final implementation of our tool is available at the Semantic Metadata Extraction repository[1].

We conduct an evaluation to assess the quality of the identified semantic metadata by comparing some tools to process the text and extract the metadata. These tools are compared to our video lecture annotator (SubAnnotator). We test the tools in different videos from computer science lectures. Our experiments examine the annotation considering

[1] https://gitlab.ic.unicamp.br/jreis/semantic-metadata-extraction

an open domain ontology (DBPedia [2]) and a specific ontology describing the computer science area [16]. Obtained results indicate that the best tools to perform the task of semantic annotating relevant terms are AutoMêta and our proposal SubAnnotator. Other assessed tools including the Ontotext and NCBO were not very effective for accomplishing this task.

The remaining parts of this document are organized in the following structure: Section 2 presents foundation concepts and discusses the related work. Section 3 describes our proposed process for the extraction of semantic metadata from video's audio as text and the way the annotator works on it. Section 4 presents the experimental evaluation methodology. Whereas Section 5 describes the experimental results of applying the different tools on the extracted text, Section 6 discusses the obtained results. Finally, Section 7 wraps up the conclusions and indicates future work.

2 Theoretical Background and Related Work

Semantic Annotation [15] refers to a process aiming to enrich a text by attaching relevant information about concepts present in the text. It links the text with formal defined concepts in some vocabulary suited to describe the meaning of the words present in the text. In this sense, a machine can process this additional information together with the text itself to understand better what the text is about. By attaching more information about some concepts from the text, we can add the context which plays a relevant role for a human to understand what a text is about. The machines do not have this context as we humans do. That is why we need to provide as much additional relevant information to the machines so that they can develop this context to help them to understand what the text is about.

One way of semantically annotating a text is through the tasks of *Named Entity Recognition* (NER) [11] and Entity Linking (EL) [12]. *Named Entity Recognition* is the process of recognizing potential parts of the text which can require some context from the reader to be better understood, such as nouns referring to a person, a company or a place. These parts are called Named Entities. Once these named entities are recognized in the text, it is possible to link them to some additional data (usually structured knowledge) which describes exactly what this entity means. This is exactly the role of Entity Linking. It uses structured or semi-structured data with a well defined semantic available from databases such as *DBPedia*[2] or *Babelnet*[3].

Semantic annotation relies on ontologies to perform the linking to semantically described concepts. Ontologies aim to represent semantics in computational systems [9] and have been designed to provide rich machine-decidable semantic representations. They refer to a formal specification of some domain, formalizing a conceptualization of a domain in terms of classes, properties and relationships between classes. Usually, a syntactic structure models the concepts of a knowledge domain, and serves as schemas that organize data expressing instances of concepts according to logical properties. The classes are the focus of ontologies because they describe the domain concepts representing groups of individuals that share

[2]https://wiki.dbpedia.org/
[3]https://babelnet.org/

properties. Data properties characterize individual attributes, whereas object properties specify the relationships between individuals of the same or different classes.

There are lots of ways to represent an ontology. One of the most common one is using the W3C Web Ontology Language[4] (OWL). It is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be exploited by computer programs. It is also part of the W3C's Semantic Web technology stack, which includes the Resource Description Framework (RDF) and the Resource Description Framework Schema (RDFS).

Resource Description Framework[5] (RDF) is a standard model for data interchange on the Web. It has features that facilitate data merging even if the underlying schemas differ. RDF specifically supports the evolution of schemas over time without requiring all the data consumers to be changed. It extends the linking structure of the Web to use URIs to name the relationship between things (predicate) as well as the two ends of the link.

The RDF data model is based on the idea of making statements about resources (in particular web resources) in expressions of the form subject–predicate–object, known as triples. The subject denotes the resource, and the predicate denotes traits or aspects of the resource, and expresses a relationship between the subject and the object. Using this simple model allows structured and semi-structured data to be mixed, exposed and shared across different applications. This linking structure forms a directed, labeled graph, where the edges represent the named link between two resources, represented by the graph nodes.

Resource Description Framweork Schema[6] (RDFS) is a general-purpose language for representing simple RDF vocabularies on the Web. It is used as base for other technologies, like OWL or SKOS, and provide language for defining structured, Web-based ontologies which enable richer integration and interoperability of data among descriptive communities.

Semantic annotation links parts of a natural language text to concepts defined in ontologies described with RDFS or OWL. As an example, Figure 1 shows part of a text from the CNN Money website[7] after the named entity recognition made by the FOX[8] tool and the entity linking to the DBPedia resources made by the AGDISTIS[9] tool. The entities recognized by the FOX NER tool are inside the square brackets in blue: The "Trump Organization" and "Ritz-Carlton". The AGDISTIS tool correctly linked these entities to the *DBPedia* resources which enable to describe their semantics explicitly.

Our investigation identified software tools for semantic annotating texts such as *AutôMeta* [1], CSO-Classifier [16], NCBO [10] and OntoText [13].

AutôMeta (automatic meta data annotation) [1] is an environment for semi-automatic (or automatic) annotation and meta-annotation of documents for publishing on the Web using RDFa, a W3C recommended annotation language based on the RDF. AutôMeta includes a RDFa extraction tool to provide the user with a view of the annotated triples.

[4] https://www.w3.org/2001/sw/wiki/OWL
[5] https://www.w3.org/RDF/
[6] https://www.w3.org/2001/sw/wiki/RDFS
[7] http://money.cnn.com/2017/08/09/news/companies/trump-golf-jupiter-appeal/index.html
[8] http://fox-demo.aksw.org/index.html#!/demo
[9] http://agdistis.aksw.org/demo/

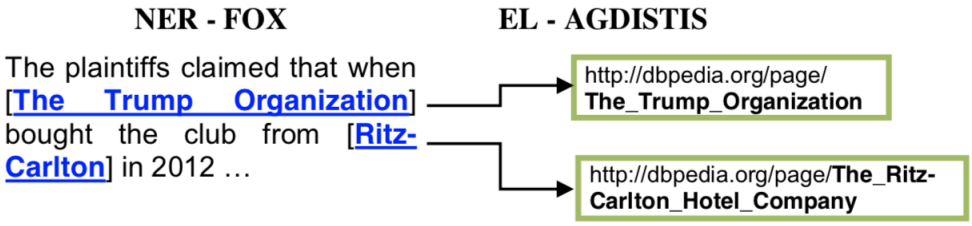

Figure 1: Example of named entity recognition and entity linking [14]

It is available in both CLI (Command Line Interface) and GUI (Graphical User Interface) interfaces. It takes the text and an ontology as input and then annotates the text based on that ontology.

CSO-Classifier [10] is a script that classifies content from scientific papers with the topics of the Computer Science Ontology [16] (CSO). Being able to synthesize the content of papers, allows to perform different kinds of analytics such as trend analysis, recommender systems, find authors' topics of interest and topic analysis. It also accepts a text to be annotated as input.

The National Center for Biomedical Ontology Annotator (NCBO) [10] is an ontology-based web service for annotation of textual biomedical data with biomedical ontology concepts. The biomedical community can use the Annotator service to tag datasets automatically with concepts from more than 200 ontologies coming from the two most important set of biomedical ontology terminology repositories: the UMLS Metathesaurus and NCBO BioPortal.

OntoText [13] is a tagging service to enrich content by pasting an URL or a piece of text. It is based on data from DBpedia and WikiData. This tool explores machine learning algorithms to recognize mentions of entities such as Person, Organisation, Location, keyphrases, and relationships between them, as well as their relevance and confidence to the text.

Santos *et al.* [7] work defines a environment for the extraction of semantic metadata from soccer games videos. It uses both semantic annotation and ontologies to do the transcription and to automatically classify the videos. Coelho *et al.* [8] define a framework to index and classify the videos lectures based on semantic annotating the video's transcription using ontologies to match the annotated terms. In their work, semantic annotation is used to create tags which describe the topics that are discussed in the video. Then, these tags are used to search the video lectures as a recommendation system. Both of these works uses the DBPedia onotology to accomplish the semantic annotation task.

[10] https://github.com/angelosalatino/cso-classifier

3 Semantic Metadata Extraction Process

Figure 2 presents the proposal for extracting the metadata and finding its occurrences in the video subtitles. In the proposed methodology to extract metadata from a video lecture posted on *YouTube*, the tool first receives as input a URL from a *YouTube* video lecture (Figure 2-a). Videos uploaded in *YouTube* were used because it automatically generates subtitles for the videos and provides an API to extract the subtitles as caption files on the SubRip format.

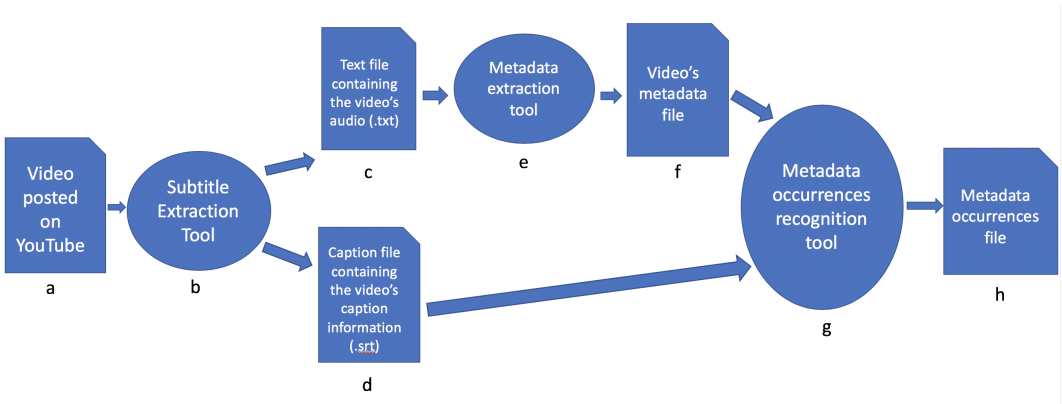

Figure 2: Our defined steps to extract the metadata

We developed the subtitle extraction software tool (Figure 2-b). It gets a YouTube's video URL as input and uses the *YouTube API*[11] to download the video's automatically generated subtitle. The subtitle is translated to the English language (if needed when the original subtitles are in Portuguese language) and the tool generates a *SubRip*[12] file. This subtitle file (Figure 2-d) contains information needed for the subtitle to be displayed while the video is playing. Listing 1 presents the beginning of a downloaded subtitle file.

```
1
00:00:02,740 --> 00:00:08,320
 [Music]

2
00:00:10,780 --> 00:00:14,809
 Hello guys

3
00:00:12,590 --> 00:00:16,040
 We're good here starting the

4
```

[11]https://developers.google.com/youtube/v3/docs/captions
[12]https://en.wikipedia.org/wiki/SubRip

```
14 00:00:14,809 --> 00:00:19,160
15  discipline of organization of
16
17 5
18 00:00:16,040 --> 00:00:21,439
19  computers and nothing is fairer than
```

Listing 1: Example of the text file created from parsing the SubRip file

This file is formatted by blocks of four lines. The first line contains an integer that represents the position of that phrase on the subtitle (first, second, third, etc); the second one is a time interval; the third is the phrase displayed on the subtitle during that time interval; and the fourth line is just a blank line.

The tool parses the subtitles to extract only the text that represents the video's audio and creates a text file (Figure 2-c). Listing 2 presents the result of parsing the SubRip file presented by listing 1.

```
1  [Music]
2  Hello guys
3  We're good here starting the
4  discipline of organization of
5  computers and nothing is fairer than
```

Listing 2: Example of the text file created from parsing the SubRip file

The subtitle extraction tool was based on the samples presented on the YouTube's github [6]. It leverages the *YouTube*'s API to download the subtitle. It also uses the API's automatic translation option to get the subtitles in the English language when the video's audio language is Portugues. Algorithm 1 presents the procedure implemented in our software tool to extract the subtitles.

Algorithm 1: Subtitle extraction tool using the YouTube API

Require: fileName, videoId {name of the text file that will be created and youtube's video id}

1: Create an authorization object to make requests to the Youtube API
2: Create an object which will be used to make the requests to the API
3: Make a request to get information of video which the id is videoId from Youtube, including subtitle information
4: Get the id of the first caption from the video object returned by the API
5: Make a request to the Youtube API to download the English caption SRT file for that caption ID
6: Convert the SRT file to a text file
7: Save the result to a text file named fileName

The generated subtitle text file is used as input for the Metadata Extraction Tool (Figure 2-e). This tool reads the input file and searches for terms that matches with concepts from

an ontology. In this work, the expression *term* is used to refer to either a single word or a few consecutive words contained within the input text file. The expression *concept* is used to refer to either a single word or a set of words that is contained within an ontology. The semantic annotation process recognizes the concept (from the ontology) in the input free-text from the subtitle. The text input file consists in a set of terms and an ontology consists in a set of concepts and relationships among them.

The list of matched terms are the video's metadata (relating the text terms with the ontology concepts) as the output of the metadata extraction tool (Figure 2-f). In this work, we consider and evaluate the application of more than one semantic annotation tool to extract the metadata. Each metadata extraction tool outputs its results in a particular file format. A common one is the *RDFa*[13]. Listing 3 presents an example of a snippet of the video lecture's metadata file in the RDFa format connecting the text with ontology concepts.

```
1 <body>
2 [Music]
3 <br />  Hello guys
4 <br />  We're good here starting the
5 <br />
6 <span id='am-426' about='ontology:discipline' typeof='owl:
    Thing'>discipline</span>
7 of
8 <span id='am-159' about='schema:Organization' typeof='schema:
    Organization'>organization</span>
9 of
10 <br />
11 computers and nothing is fairer than
12 <br />
13 people
14 <span id='am-183' about='ontology:start' typeof='owl:Thing'>
    start</span>
15 talking about this
16 <span id='am-41' about='schema:Product' typeof='schema:
    Product'>product</span>
17 <br />
18 digital called computer to computer
```

Listing 3: Video's Lecture Metadata file example

Aiming to obtain a baseline for our experimental evaluations (cf. Section 4), we defined our metadata extraction tool, named as SubAnnotator. Algorithm 2 describes the SubAnnotator. It takes two files as input: the first is a text file containing a list of concepts that are candidates to be annotated (from an ontology or set of ontologies). The second input contains the text which is meant to be annotated, such as presented in Figure 2-c. The

[13] https://rdfa.info/

algorithm searches for the ontology's concepts in the input text by getting all of the subsets of one, two, three or four consecutive words (input text snippet) and checking if this subset of words matches to any of the ontology's concepts present on the list of concepts provided as input. The *countWords* function returns the number of words in the input text. The *extractSnippet* function extract a subset of *termSize* consecutive words from the text starting from the ith word in the text. The *add* function adds the candidate to the *annotatedTerms* list. The algorithm returns a list of all matched terms, as presented by Figure 2-f.

Algorithm 2: SubAnnotator

```
Require: conceptsList, text
1: textSize ← countWords(text)
2: annotatedTerms ← ∅
3: for termSize ← 1 to 4 do
4:    for i ← 1 to (textSize - termSize) do
5:       candidate ← extractSnippet(text, i, termSize)
6:       if conceptsList contains candidate then
7:          annotatedTerms ← add(candidate)
8:       end if
9:    end for
10: end for
11: return annotatedTerms
```

Two concepts files were used as input for the SubAnnotator. The first one contains the list of concepts from the DBPedia ontology and the second one contains the concepts from the Computer Science ontology. We developed a parser that receives the ontology file as input and obtains the list of concepts for that ontology which will be used by the SubAnnotator as input. The parser receives the ontology file as input and scans through it searching for the English label for each class definition. The ontology's concepts are the contents of these labels. Algorithm 3 presents the developed parser. The function *isEOF* returns a boolean indicating if all the lines on the file has been read or not. The function *nextLine* reads the next line from the file and returns it. The function *substring* gets a string as first argument and returns the substring starting from the character that follows the occurrence of the character that is passed as second argument and the character that precedes the first occurrence of the character that is passed as third argument after the first occurrence of the character that is passed as second argument. The *add* function receives an argument and adds it to the *conceptsList*.

The DBPedia ontology is described using the Resource Definition Framework Schema [14] (RDFS). Listing 4 shows a snippet of a concept definition from this ontology. For each ontology concept, it first defines an owl:Class with the rdf:about attribute that contains the URL for the definition of that concept. Then it defines the rdfs:label which are the text representation for that concept. Each label is attached to a language that is defined using the xml:lang attribute. The rdfs:subClassOf tag is used to represent relationship between

[14] https://dvcs.w3.org/hg/rdf/raw-file/default/rdf-schema/index.html

concepts.

```
<owl:Class rdf:about="http://dbpedia.org/ontology/Tax">
  <rdfs:label xml:lang="de">Steuer</rdfs:label>
  <rdfs:label xml:lang="es">impuesto</rdfs:label>
  <rdfs:label xml:lang="fr">taxe</rdfs:label>
  <rdfs:label xml:lang="en">tax</rdfs:label>
  <rdfs:label xml:lang="nl">belasting</rdfs:label>
  <rdfs:label xml:lang="el">&#x3C6;&#x3CC;&#x3C1;&#x3BF;&#
 x3C2;</rdfs:label>
  <rdfs:label xml:lang="ja">&#x79DF;&#x7A0E;</rdfs:label>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/
 owl#Thing"/>
  <rdfs:subClassOf rdf:resource="http://www.
 ontologydesignpatterns.org/ont/dul/DUL.owl#Description"/>
  <prov:wasDerivedFrom rdf:resource="http://mappings.
 dbpedia.org/index.php/OntologyClass:Tax"/>
```

Listing 4: Snippet of the DBPedia ontology

Algorithm 3: DBPedia's concepts Parser

Require: dbpediaOntology

```
conceptsList ← ∅
while isEOF() == false do
    line ← nextLine()
    if line contains the string "<rdfs:label xml:lang="en">" then
        concept ← substring(line,'>','<')
        conceptsList ← add(concept)
    end if
end while
return conceptsList
```

The Computer Science ontology is described using the Resource Definition Framework [15] (RDF). Listing 5 shows a snippet of a concept definition for the ontology. It starts with a rdf:Description tag definition with the rdf:about attribute. This tag describes the concept it is defining and the rdf:about attribute contains an URL that defines the concept itself. It also contains a rdf:label tag which is used to provide a human-readable version of the concept and finally it defines a rdf:type tag that indicates to which class this concept is an instance of.

```
<rdf:Description rdf:about="http://cso.kmi.open.ac.uk/topics/
    wireless%20technologies">
```

[15] https://www.w3.org/RDF/

```
2         <label xmlns="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://cso.kmi.open.ac.uk/topics/wireless%20
    technologies"/>
3         <rdf:type rdf:resource="http://technologies.kmi.open.
    ac.uk/rexplore/ontologies/BiboExtension/BiboExtension.owl#
    Topic"/>
4 </rdf:Description>
```

Listing 5: Snippet of the Computer Science ontology

The parser for the Computer Science ontology works similar to the Algorithm 3 by receiving the ontology file as input and scanning through it searching for the RDF label for each description definition. The ontology's concepts are extracted from the rdf resource property of the label. The resource property contains a URL which corresponds to the concept definition and the parser interprets the URL to extract the concept label.

After running the ontologies' parsers, the list of concepts for the DBPedia ontology and for the Computer Science ontology were created and used as input for the SubAnnotator. The last step of our proposed process (Figure 2) is to run the metadata occurrences recognition tool (Figure 2-g) also developed in this work.

In this final step, the tool gets the video's metadata and the subtitle SubRip as input. Then, it obtains the list of terms from the video lecture's metadata. For each term, it searches on the SubRip subtitle file to detect the term occurrence in an exact point in time. The tool's output is the metadata occurrences embedded in the subtitles (Figure 2-h). This contains information about each term in addition to the time and position it occurs on the subtitle. Listing 6 presents an example of the final result. Each line contains four information in the following order: the metadata itself; the time interval which it is displayed on the subtitle; the phrase that is displayed on the subtitle during that time interval and the position of that phrase in the subtitle file (if it is the first, second, third phrase to appear in the subtitle).

```
1 discipline 00:00:14,809 --> 00:00:19,160  discipline of
    organization of  4
2 organization 00:00:14,809 --> 00:00:19,160  discipline of
    organization of  4
3 start 00:00:19,160 --> 00:00:24,230  people start talking
    about this product  6
4 product 00:00:19,160 --> 00:00:24,230  people start talking
    about this product  6
```

Listing 6: Metadata Occurrences example

4 Experimental Evaluation

This evaluation aimed to assess the quality of the results obtained by the proposed semantic metadata extraction process. To this end, the first step was to define which available Semantic Annotation tools would be used as the metadata extraction tool, as a way of comparing with the SubAnnotator.

The features considered during this selection were the following: 1) API exposure because this work aims at providing an available implementation of the process, so it is important that the tool can be used programmatically by exposing some kind of API. In this sense, it is possible to run as many tests as desired; 2) Flexibility on the used ontology. As the semantic annotation tools are only able to annotate terms which matches the used ontology's concepts, if the tool supports more than one ontology it is possible to test different ontologies and determine which one is the best suited for annotating videos in a specific subject. We considered the following external annotation tools:

1. *AutôMeta* [1] was the first choice because it enables to define which ontology is used in the annotation. *AutôMeta* can be run programmatically as a java program.

2. The CSO-Classifier was chosen because it explores an Computer Science Ontology [16], released very recently. CSO-Classifier enables us to compare its results to the other tools that uses such Computer Science Ontology as well. This tool provides an API so it is possible to automate the tests.

3. NCBO [10] was another chosen annotation tool because it provides an API [16] to run the tests. It accepts an ontology from its database as input. It was not possible to use the Computer Science Ontology [16] with the NCBO annotator.

4. OntoText was our last choice. It provides an API [17] and explores the DBPedia ontology for its annotation. This aspect makes it a good option to be compared with other tools using the same ontology.

In addition to these external options, our SubAnnotator tool was used to run the tests.

4.1 Datasets

The second step was to define the used ontologies in the evaluation. The first choice was the DBPedia ontology [2]. It is a shallow, cross-domain ontology, which has been manually created based on the most commonly used infoboxes within Wikipedia. It contains more than 3000 classes that was used as the ontology's concepts in this work. The second choice was the Computer Science Ontology [16] because the videos chosen to run the initial tests are about the computer science area. Therefore, it is expected that the quality of the annotations be better for this ontology rather than for the DBPedia one.

The third step was to define a set of video lectures to test the tools. To this end, we chose two disciplines in computer science. The first one was Computer Architecture and

[16] https://github.com/ncbo/ncbo_rest_sample_code
[17] https://tag.ontotext.com/documentation/

the second one was Data Structures. For the first discipline, we selected three videos: two in Portuguese language and the other in English language. For the second discipline, we chose four videos: two in Portuguese and two videos in English language. Although a few Portuguese videos were selected, our proposal explored the *YouTube*'s automatic subtitle translation. All the downloaded subtitles were in the English language. Table 1 shows information about the chosen videos.

ID	Topic	Language	Duration	Number of Words	YouTube's ID
1	Computer Architecture	Portuguese	1:38:14	5858	2PtkVHCF1eE
2	Computer Architecture	Portuguese	0:35:21	5056	dQ6dzOPY9uc
3	Computer Architecture	English	0:13:53	2361	So9SR3qpWsM
4	Data Structures	Portuguese	0:40:16	6065	G0OKWQN9Jt4
5	Data Structures	Portuguese	0:33:34	4943	FzPceEhQCSQ
6	Data Structures	English	0:52:39	7129	9Jry5-82I68
7	Data Structures	English	0:52:31	6739	B7hVxCmfPtM

Table 1: Information about the videos used to evaluate the annotation tools

The video's ID is used as the video identifier within this document. The topic refers to which computer area the video talks about. Language refers to the original video's audio language. Number of words refers to the number of words of the subtitle's text (Figure 2-c). The youtube's ID can be used to get the URL to the video[18] itself. Section 5 presents results for the annotation quality organized by each one of these videos.

4.2 Tools Setup

After defining all tools and datasets, we setup the annotation software tools as a preparation step to run the tests. In the following, we present the setup aspects for each considered annotation tool.

4.2.1 AutoMêta

In order to run the *AutoMêta*, it was needed to setup which ontology it uses. It was possible to choose to run the tool with a reasoner or not. The available reasoners are Pellet [19] and Hermit [20]. For this tool we selected two setups. The first one was to run it with the DBPedia ontology and the second one was to run it with the Computer Science ontology. No reasoner was used for both setups because when running with the computer science ontology, the tool was unable to use any reasoner. In order to keep consistency to analyze the results, no reasoner was used for both evaluation tests.

[18] https://www.youtube.com/watch?v="add-youtube's-id-here"
[19] https://www.w3.org/2001/sw/wiki/Pellet
[20] http://www.hermit-reasoner.com/

4.2.2 CSO-Classifier

The only setup needed for running the CSO-Classifier was defining which version of the Computer Science Ontology [16] should be used. In our experimental tests, we selected only one setup for this tool which used the second ontology's version that has around twenty-six thousand concepts.

4.2.3 NCBO

NCBO has a series of setup parameters. For example, whether it should remove stop words or not, whether it should consider only whole words or not, whether it should exclude numbers or not, among others. The list of parameter can be found on the API documentation [21]. In our experimental tests, only one setup was used for this tool. It used the default values for all the parameters except for the ontology one, which was set to be the Computer Network ontology [22] which has about five hundreds classes. The decision of using this ontology was to try another ontology related to Computer Science that were not so specific to test the influence of the ontologies on the results.

4.2.4 OntoText

OntoText was the simplest tool used because it only took the text to be annotated as input. It uses the DBPedia ontology as the basis to obtain the annotations. Therefore only setup for this tool is using the DBPedia ontology.

4.2.5 SubAnnotator

Our SubAnnotator only required a list of the concepts which can be annotated besides the text that should be annotated. Two setups were selected for this tool. The first one explored the concepts from the DBPedia and the second one used the concepts from the Computer Science ontology.

Therefore, a total of seven setups (experimental configurations) were used to conduct the evaluation (as a metadata extraction tool option in the Figure 2-e):

1. AutoMêta with DBPedia ontology
2. AutoMêta with Computer Science ontology
3. CSO-Classifier with the Computer Science ontology
4. NCBO with the Computer Network ontology
5. Ontotext with DBPedia ontology
6. SubAnnotator with DBPedia ontology

[21] http://data.bioontology.org/documentationnav_annotator
[22] https://bioportal.bioontology.org/ontologies/CN

7. SubAnnotator with Computer Science ontology

After defining the tool's setups, the experimental tests were run. For each selected video, every tool setup was tested by using the video as input. Therefore, for each video there were seven results to be analyzed.

4.3 Metrics

This investigation aimed to analyze the effectiveness of these tools on the entity recognition task in the context of video lectures annotation. This section defines the metrics used for the results analysis.

We defined a set of terms that would describe each video well in order to conduct the result analysis. For this purpose, we build this set by running all the tools for each video and get all the distinct annotated words. Then, this set of all distinct annotated words was manually scanned. Those terms considered irrelevant was removed from the set. The remaining terms represent the set of **relevant terms**. Its size for each video is represented by the letter DRV in this investigation.

In order to make the metric's understanding easier, we use some examples for illustration purpose. Consider the listings 7 as the set of relevant terms for a video (as an example):

```
[assembly, basic block, compiler, memory, operating systems]
```

Listing 7: Example of set of relevant terms.

There are 5 relevant terms in the set; thus the size of the set of relevant terms for that video is 5 ($DRV = 5$). Also, consider the listing 8, which represents all the terms that some tool has annotated.

```
[assembly, activity, assembly, address, basic block, book,
    center, compiler, basic block, series, compiler, state,
    memory, programming, result, resolution, range, number]
```

Listing 8: List of all annotated terms by some tool.

4.3.1 Precision (Pr)

It is the fraction of distinct relevant terms that were correctly annotated by the tool among all distinct annotated terms. It represents the precision of the tool on annotating relevant terms.

$$Pr = DRA/DTA \tag{1}$$

DRA = number of distinct (without repetition) relevant terms that the tool annotated correctly (according to the set of relevant terms)

DTA = total number of distinct (without repetition) terms that the tool annotated (it includes the terms that are not relevant according to the set of relevant terms)

For the given example, the number of distinct relevant terms that the tool annotated is four (DRA = 4) because the tool was able to annotate four relevant terms in a correct way: "assembly", "basic block", "compiler" and "memory". Those terms are all part of the set of relevant terms given as example. The total number of distinct terms that the tool annotated is fifteen (DTA = 15). These terms are: "assembly", "activity", "address", "basic block", "book", "center", "compiler", "series", "state", "memory", "programming", "result", "resolution", "range" and "number". Therefore the tool precision for the video example is 0.27 (Pr = 4/15 = 0.27)

4.3.2 Recall (Re)

It is the fraction of distinct relevant terms that were correctly annotated among all distinct relevant terms for that video. It represents the tool's ability to recognize the relevant terms for that video.

$$Re = DRA/DRV \tag{2}$$

DRA = number of distinct relevant terms that the tool correctly annotated

DRV = total number of distinct relevant terms for that video

For the given example, the number of distinct relevant terms that the tool annotated is four (DRA = 4) because the tool was able to correctly annotate four relevant terms: "assembly", "basic block", "compiler" and "memory". The total number of distinct relevant terms for the video example is five (DRV = 5) because there are five terms in the set of relevant terms (expected set): "assembly", "basic block", "compiler", "memory" and "operating systems". Therefore, the tool's recall for the video is 0.8 (Re = 4/5 = 0.8).

4.3.3 F-Measure (F)

It is the harmonic mean of precision (Pr) and recall (Re). It is used to rank the tools in our result analysis.

$$F = 2 * (Pr * Re)/(Pr + Re) \tag{3}$$

Pr = Precision

Re = Recall

4.3.4 Total Number of Relevant Terms Annotated (TRA)

We conducted a further quantitative analysis for each tool. The more relevant terms the tool can annotate the more metadata information it can provide. Whereas considering the **distinct** annotated terms (without repetition) is useful for a qualitative analysis, this is

still unable for considering the total number of relevant terms annotated by the tool. We define the metric TRA for this purpose.

For the given example, the total number of relevant terms annotated is seven (TRA = 7). The tool was able to annotated the term "assembly" twice (1st and 3rd annotated terms), the term "basic block" twice (5th and 9th annotated terms), the term "compiler" twice (8th and 11th annotated terms) and the term "memory" once (13th annotated term).

5 Results

The results are presented as tables for each test concerning the videos defined in Table 1. The results presented in the tables are in descending order by the values of the F-Measure. Since the results are highly dependent on the used set of relevant terms, for each table, we present the list of relevant terms considered for that video. After each individual video's analysis, we present a table with the overall result (combination of results for all videos). Section 6 discusses the obtained findings.

5.1 Video 1 (Computer Architecture - Portuguese Language)

Table 2 presents the results for video 1.

Tool's Name	**Ontology**	**DRV**	**DTA**	**DRA**	**Pr**	**Re**	**F-Measure**	**TRA**
SubAnnotator	DBPedia	60	93	44	0,47	0,73	0,58	166
AutoMêta	DBPedia	60	106	47	0,44	0,78	0,57	89
AutoMêta	Computer Science	60	22	12	0,55	0,20	0,29	15
Ontotext	DBPedia	60	49	11	0,22	0,18	0,20	36
SubAnnotator	Computer Science	60	9	6	0,67	0,10	0,17	9
CSO-Classifier	Computer Science	60	9	6	0,67	0,10	0,17	9
NCBO	Computer Network	60	3	0	0,00	0,00	0,00	0

Table 2: Results for video 1

List of relevant terms for video 1:

`[activity, added, address, area, assembly, average, background, basic block, block, book, center, class, code, compile, compiler, computer, concept, convention, curve, destination, different, dna, event, feature, field, file, files, formula, integer, language, lowest, material, memory, model, operating system, order, position, power, procedure, procedures, programming language, queue, radio, range, region, resolution, result, series, software, stack, state, storage, string, subtraction, symbol, time, type, value, version, work]`

For this video, the SubAnnotator and the AutoMêta using the DBPedia ontology reveled the best results. Although the precision was better overall for the tools that used the Computer Science Ontology, the recall was low for them, which affected the F-Measure value. Considering the total number of relevant annotated words, the SubAnnotator was

able to annotate 166 words whereas the AutoMêta was able to annotate 89, which shows that AutoMêta missed some words that could have been annotated. NCBO had the poorest quality of annotation being able to annotate only 3 distinct words and none of them were relevant.

5.2 Video 2 (Computer Architecture - Portuguese Language)

Table 3 presents the results for video 2.

Tool's Name	**Ontology**	**DRV**	**DTA**	**DRA**	**Pr**	**Re**	**F-Measure**	**TRA**
AutoMêta	Computer Science	106	50	44	0,88	0,42	0,56	50
AutoMêta	DBPedia	106	99	56	0,57	0,53	0,55	113
SubAnnotator	Computer Science	106	47	41	0,87	0,39	0,54	156
CSO-Classifier	Computer Science	106	45	39	0,87	0,37	0,52	139
SubAnnotator	DBPedia	106	79	45	0,57	0,42	0,49	198
Ontotext	DBPedia	106	36	16	0,44	0,15	0,23	164
NCBO	Computer Network	106	14	6	0,43	0,06	0,10	29

Table 3: Results for video 2

List of relevant terms for video 2:

`[Transaction Processing, access, activity, added, address, area, artificial intelligence, associated with, autonomy, band, binary code, bluetooth, board, book, capacity, cell phones, central processing unit, class, code, command, comment, compiler, component, computer, computer hardware, computer science, concept, configuration, cpu, decision systems, definition, destination, device, device driver, different, display, embedded, embedded system, embedded systems, energy, equipment, features, file, format, frequency, hardware, hardware components, information, information systems, infrared, interconnection, internet, interpreter, investments, keyboard, kinect, language, languages, linux, mac, management, manager, material, memory, microcontroller, microwave, mode, multi-core, multimedia, network, operating system, operating systems, organization, os, output, parent, port, power, product, programming language, purpose, radio, result, scale, single, smartphone, smartphones, social network, software, source, source code, syntax, time, version, video card, weight, wifi, wireless, wireless network, work, binary codes, bus, buses, OS, cards, intel]`

For this video, the F-Measure values were similar for the AutoMêta, SubAnnotator and the CSO-Classifier. The precision was higher for those which used the Computer Science ontology and the recall was higher for those which used the DBPedia ontology. The highest recall was for the AutoMêta with the use of the DBPedia ontology: 56 distinct relevant annotated terms out of 106. Regarding the total number of relevant annotated words, the SubAnnotator was the one which was able to annotated the highest number of terms. Although Ontotext was able to annotate a high number of relevant terms and its precision

was good, its recall was considerably low: only 16 terms recognized out of 106. NCBO had the worst results out of all tools.

5.3 Video 3 (Computer Architecture - English Language)

Table 4 presents the results for video 3.

Tool's Name	Ontology	DRV	DTA	DRA	Pr	Re	F-Measure	TRA
SubAnnotator	Computer Science	35	16	15	0,94	0,43	0,59	56
CSO-Classifier	Computer Science	35	15	14	0,93	0,40	0,56	53
AutoMêta	Computer Science	35	13	11	0,85	0,31	0,46	11
SubAnnotator	DBPedia	35	24	10	0,42	0,29	0,34	26
AutoMêta	DBPedia	35	27	9	0,33	0,26	0,29	18
NCBO	Computer Network	35	7	5	0,71	0,14	0,24	13
Ontotext	DBPedia	35	5	0	0,00	0,00	0,00	0

Table 4: Results for video 3

List of relevant terms for video 3:

`[cluster, code, purpose, CPU, scale, power, location, device, operating system, order, microprocessor, keyboard, peripheral, memory, multiprocessing, computer, multiprocessor, throughput, communication, computer system, computer resources, general purpose, high availability, servers, system architecture, computer systems, multiprocessor system, multiprocessor, binary code, cluster systems, communication, CPU, architecture, binary code, cluster systems, communication, computer resources, computer system, general purpose, high availability, keyboard, microprocessor, multiprocessor, operation system, server, bus, performance, scale, synchronization, throughput]`

For this video, the tools which used the Computer Science ontology had the best results. The precision was high for all of those tools and the highest recall was not excellent: 15 distinct relevant annotated terms out of 35. Regarding the total number of relevant annotated terms, SubAnnotator obtained the best result again: 56. Although the AutoMêta with the DBPedia ontology was the one which had the highest number for the distinct annotated terms, its precision was low. NCBO had bad results being able to annotate only seven distinct terms. We can consider the influence of the used ontology for this finding. Ontotext had the worst result being able to annotate only 5 distinct terms and none of them were relevant.

5.4 Video 4 (Data Structures - Portuguese Language)

Table 5 presents the results for video 4.

Tool's Name	Ontology	DRV	DTA	DRA	Pr	Re	F-Measure	TRA
AutoMêta	DBPedia	37	90	33	0,37	0,89	0,52	52
SubAnnotator	DBPedia	37	73	27	0,37	0,73	0,49	177
AutoMêta	Computer Science	37	12	5	0,42	0,14	0,20	6
SubAnnotator	Computer Science	37	5	2	0,40	0,05	0,10	14
CSO-Classifier	Computer Science	37	5	2	0,40	0,05	0,10	14
Ontotext	DBPedia	37	11	1	0,09	0,03	0,04	69
NCBO	Computer Network	37	9	0	0,00	0,00	0,00	0

Table 5: Results for video 4

List of relevant terms for video 4:

`[abstract, access, added, boolean, builder, capacity, child, closed, code, computer, construction, cost, definition, end point, field, head, information, insert, integer, java, language, limit, listen, memory, mother, movement, position, project, second, sentence, service, space, start, state, third, time, value]`

Results in Table 5 indicate that AutoMêta and SubAnnotator with the DBPedia ontology obtained the best results. Although the precision was low for them (0.37), the recall was considerably good (0.89 for AutoMêta and 0.73 for SubAnnotator). Although Ontotext used the same ontology, it was able to annotated only one out of 37 distinct relevant terms making its recall really bad. Regarding the total number of relevant annotated terms, SubAnnotator got 177, Ontotext 69 and AutoMêta 52. It shows again that AutoMêta does not annotated all possible terms. For the tools that used the Computer Science ontology, the recall was really bad because they were able to annotated a maximum of 5 distinct relevant terms out of the 37 possible. NCBO got the worst results one more time because it was not able to annotated any relevant term.

5.4.1 Video 5 (Data Structures - Portuguese Language)

Table 6 presents the results for video 5.

Tool's Name	Ontology	DRV	DTA	DRA	Pr	Re	F-Measure	TRA
AutoMêta	DBPedia	32	69	24	0,35	0,75	0,48	35
SubAnnotator	DBPedia	32	58	20	0,34	0,63	0,44	81
AutoMêta	Computer Science	32	18	6	0,33	0,19	0,24	6
SubAnnotator	Computer Science	32	8	3	0,38	0,09	0,15	8
CSO-Classifier	Computer Science	32	8	3	0,38	0,09	0,15	8
Ontotext	DBPedia	32	23	4	0,17	0,13	0,15	51
NCBO	Computer Network	32	6	1	0,17	0,03	0,05	12

Table 6: Results for video 5

List of relevant terms for video 5:

`[approach, binary tree, child, children, circle, comment, database, decision-making, father, food, head, information, insert, kind, license, list, memory, model,`

`order, parent, queue, requirement, result, review, root, single, start, supply, time, tree, trees, value]`

The results for this video were really similar to those of video 4. Results in Table 6 indicate that AutoMêta and SubAnnotator with the DBPedia ontology obtained the best results. Although the precision was low for them, the recall was good. Although Ontotext used the same ontology, it was able to annotated only four out of 32 distinct relevant terms making its recall really bad. Regarding the total number of relevant annotated terms, SubAnnotator got 81, Ontotext 51 and AutoMêta 35. It shows again that AutoMêta does not annotated all possible terms. For the tools that used the Computer Science ontology, the recall was really bad because they were able to annotated a maximum of 6 distinct relevant terms out of the 32 possible. NCBO got the worst results one more time being able to annotate only one relevant term.

5.5 Video 6 (Data Structures - English Language)

Tool's Name	Ontology	DRV	DTA	DRA	Pr	Re	F-Measure	TRA
SubAnnotator	DBPedia	41	50	28	0,56	0,68	0,62	202
AutoMêta	DBPedia	41	65	31	0,48	0,76	0,58	57
Ontotext	DBPedia	41	21	7	0,33	0,17	0,23	198
SubAnnotator	Computer Science	41	6	5	0,83	0,12	0,21	51
CSO-Classifier	Computer Science	41	6	5	0,83	0,12	0,21	51
AutoMêta	Computer Science	41	9	5	0,56	0,12	0,20	5
NCBO	Computer Network	41	4	1	0,25	0,02	0,04	32

Table 7: Results for video 6

List of relevant terms for video 6:

`[BST, access, added, array, binary search tree, binary tree, building, child, definition, existence, free, hash, heap, height, high quality, information, insert, intermediate node, kind, left child, license, list, location, max, min, operator, optimization, order, parent, procedure, quality, rank, requirement, right child, second, single, start, time, tree, value, version]`

Results in Table 7 indicate that the tools which used the DBPedia ontology had better results. From the three that composes this category, Ontotext was the one which got the worse results. Both its precision (0.33) and recall (0.12) was low. However, it was able to annotate way more occurrences of relevant terms than AutoMêta (198 against 57). SubAnnotator obtained the best results and it was able to annotate the most occurrences of relevant terms (202). The tools that used the Computer Science ontology had similar f-measure results. They got a very good precision overall. Nevertheless, its recall was considerably low with a maximum of 0.12. NCBO got the worst result again, being able to annotate only one relevant term.

5.6 Video 7 (Data Structures - English Language)

Table 8 presents the results for video 7.

Tool's Name	Ontology	DRV	DTA	DRA	Pr	Re	F-Measure	TRA
SubAnnotator	DBPedia	41	45	29	0,64	0,71	0,67	176
AutoMêta	DBPedia	41	57	33	0,58	0,80	0,67	53
CSO-Classifier	Computer Science	41	7	5	0,71	0,12	0,21	32
SubAnnotator	Computer Science	41	8	5	0,63	0,12	0,20	32
AutoMêta	Computer Science	41	12	5	0,42	0,12	0,19	5
Ontotext	DBPedia	41	15	3	0,20	0,07	0,11	59
NCBO	Computer Network	41	3	1	0,33	0,02	0,05	19

Table 8: Results for video 7

List of relevant terms for video 7:

`[Big O, abstract, binary tree, block, building, child, code, definition, description, free, head, height, high quality, insert, kind, left child, license, limit, list, max, min, n log n, order, parent, power, priority queue, procedure, produces, quality, queue, result, right child, route, second, series, single, start, time, tree, value, visualization]`

For the last video, the results were similar to those of video 4. The best tool were the SubAnnotator and AutoMêta with the DBPedia ontology. SubAnnotator was able to annotate more occurrences of relevant terms than AutoMêta (176 against 53). The tools that used the Computer Science ontology had bad recall (0.12 for all of them). Ontotext was able to annotate only 3 relevant terms while NCBO was able to annotate only one.

5.7 Overall Analysis

Table 9 presents the results to analyze the overall effectiveness for each setup. The values for DRV, DTA, DRA and TRA are the sum of the seven individual values each setup obtained for each video. We calculated the overall precision, recall and f-measure for each setup. The results shows that AutoMêta and SubAnnotator with the DBPedia ontology got the highest recall overall (0.66 and 0.58 respectively). It also shows that the precision for the tools that used the Computer Science ontology were the highest (0.65 for AutoMêta, and 0.78 for both SubAnnotator and CSO-Classifier). However, the recall for these tools were low which made their overall f-measure low. Regarding Ontotext and NCBO, both tools had really low recall (0.12 and 0.04 respectively) and the lowest precision among all tools (0.26 and 0.30 respectively). Considering the total number of relevant terms annotated, SubAnnotator with the DBPedia ontology had the best performance by annotating 1026 terms. Ontotext was the second best in this metric, being able to annotate 577 relevant terms and AutoMêta with the Computer Science ontology was the tool that had the lowest result annotating only 98 relevant terms overall.

Tool's Name	Ontology	DRV	DTA	DRA	Pr	Re	F-Measure	TRA
AutoMêta	DBPedia	352	513	233	0,45	0,66	0,54	417
SubAnnotator	DBPedia	352	422	203	0,48	0,58	0,52	1026
AutoMêta	Computer Science	352	136	88	0,65	0,25	0,36	98
SubAnnotator	Computer Science	352	99	77	0,78	0,22	0,34	326
CSO-Classifier	Computer Science	352	95	74	0,78	0,21	0,33	306
Ontotext	DBPedia	352	160	42	0,26	0,12	0,16	577
NCBO	Computer Network	352	46	14	0,30	0,04	0,07	105

Table 9: Overall Results

6 Discussion

Based on the results presented in Section 5, we highlight that the ontology used by the annotation tools play an important role in the task of annotating the terms. Containing concepts that matches with relevant terms for the video leads to better results. This can be confirmed by looking at the best results for each video and comparing the ontology used for those tools.

Concerning the ontologies used in the tests, primarily, it was expected that the Computer Science ontology would give better results than the DBPedia ontology, because the videos used on the tests are videos about computer science subjects. However, the results showed that the DBPedia ontology contains concepts that are useful to describe these kind of videos as compared to the Computer Science ontology. We observed that this is due to the fact that the Computer Science ontology described broad areas of computer science, and it does not get too deep in the specific subjects. Therefore, one could expect that if there were ontologies for the specific subjects, which the videos subject are about (Computer Architecture and Data Structures), these ontologies could lead to better annotation results.

Regarding the tools, the results indicated that AutoMêta and the SubAnnotator were the ones reaching the best results. Their precision and recall were good overall as well as their ability to annotate multiple occurrences of relevant terms based on the input text. An issue that deserves consideration concerns the fact that AutoMêta is not very effective in annotating multiple occurrences of relevant terms. The results showed that SubAnnotator annotated more terms than AutoMêta (1026 *vs* 417 overall) taking into account as the input text was always the same for them. Therefore, AutoMêta should have annotated more terms as well.

Ontotext obtained the worst results even using the DBPedia ontology. It was not able to annotate many distinct terms and most of the annotated terms were irrelevant for the video. This two things made dropping drastically its precision and recall. However, it was the second best overall on total number of relevant terms annotated. It shows that it is good on recognizing terms from the input text, but the number of distinct relevant terms it can recognize is very low.

NCBO was the tool that had the worst results among all. Clearly the ontology used had a huge impact on its results making its overall recall extremely low (0.04) As none of the videos were about computer networks, they did not have many relevant terms in common

with the ontology's concepts. This was probably the main cause for the bad results this tool got overall.

The main challenges for the future work on extracting metadata from video lectures are related to defining an ontology that is able to hold the most relevant terms for the video. The results have shown that using broad ontologies such as the DBPedia one and the Computer Science can give good results on extracting some relevant terms from the video. However, the ideal ontology would be the one that contained all the relevant terms for each video. The results showed that none of the setups were able to annotated all the relevant terms (the recall was never 1.0). Another challenge is to get the input texts in the best quality as possible so that the terms are not misspelled and the text itself is coherent. Specially for videos which language are not English, this is a big challenge that influences in the quality of the extracted metadata. The final challenge is to find a tool that gets both good qualitative results and good quantitative results. The tests have shown that some tools are really good on extracting relevant metadata from the videos but not so good on annotating a good amount of occurrences for that terms. Other tools are effective on annotating a good amount of occurrences for a few relevant terms.

7 Conclusion

The extraction of semantic metadata in video lectures can be essential to improve the recommendation and retrieval of videos in the learning process support. In this work, we proposed a software tool to automatically extract metadata from video lectures subtitles based on some ontologies[23]. We conducted an experimental evaluation to assess the quality of the annotation obtained by our SubAnnotator, an algorithm that recognizes ontology concepts from the video subtitles free-text. Our experiments enabled the comparison of several semantic annotation tools with different ontologies. This work has helped to understand which parts of the whole metadata extraction process can influence the most on the quality of the extracted metadata, so it is possible to focus on improving them. We found that both our SubAnnotator and the AutoMêta present as adequate tools to perform the metadata extraction task. Our results indicated that the main challenge on extracting relevant metadata is defining an ontology containing concepts that describe the video's subject in an optimal way. In future work, we plan to create more specific ontologies that contain as many as relevant terms for each video's subject as possible and use them as input for the semantic annotation tools.

References

[1] Automêta - a semantic annotation tool. URL: https://github.com/celsowm/AutoMeta, last accessed on 11/12/2018.

[2] Dbpedia ontology. https://wiki.dbpedia.org/services-resources/ontology, last accessed on 11/12/2018.

[23] tool available in: https://gitlab.ic.unicamp.br/jreis/semantic-metadata-extraction

[3] otranscribe - a free web app to take the pain out of transcribing recorded interviews. URL: https://otranscribe.com/, last accessed on 11/12/2018.

[4] Trint. URL: https://trint.com/, last accessed on 11/12/2018.

[5] Youtube captions api. https://developers.google.com/youtube/v3/docs/captions, last accessed on 11/12/2018.

[6] Youtube java api sample code. URL: https://github.com/youtube/api-samples/blob/master/java, last accessed on 11/12/2018.

[7] Alexandre Santos Celso AS Santos and Tatiana A Tavares. Uma estratégia para a construção de ambientes para a descrição semântica de vídeos. In *Proc of*, page 274–281, 2007.

[8] Sandro Athaide Coelho and Jairo Francisco de Souza. Anotação semântica de transcritos para indexação e busca de vídeos. In *Conferência Ibero Americana (WWW/INTERNET)*. [GS Search], 2014.

[9] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human Computer Studies*, 43(5-6):907–928, December 1995.

[10] Clement Jonquet, Nigam Shah, Cherie Youn, Chris Callendar, Margaret-Anne Storey, and M Musen. Ncbo annotator: semantic annotation of biomedical data. In *International Semantic Web Conference, Poster and Demo session*, volume 110, 2009.

[11] Dan Roth Lev Ratinov. Design challenges and misconceptions in named entity recognition. In *Conference on Computational Natural Language Learning (CoNLL)*, page 147–155. Association for Computational Linguistics, 2009.

[12] Dan Roth Lev Ratinov. Glow tac-kbp 2011 entity linking system. In *TAC*. Text Analysis Conference, 2011.

[13] Bernardo Magnini, Matteo Negri, Emanuele Pianta, Lorenza Romano, Manuela Speranza, Luciano Serafini, Christian Girardi, Valentina Bartalesi, and Rachele Sprugnoli. From text to knowledge for the semantic web: the ontotext project. In *SWAP*, volume 166, 2005.

[14] Jean Carlos Oliveira de Abreu. Caracterização e tratamento de problemas de casamentos parciais no reconhecimento de menções em textos. Master's thesis, Universidade Federal de Santa Catarina, 2018.

[15] Scerri S Handschuh S Sintek M Oren E, Möller K. What are semantic annotations? 2006.

[16] Mannocci A. Osborne F. Motta E. Salatino A.A., Thanapalasingam T. The computer science ontology: A large-scale taxonomy of research areas. Springer, Cham, September 2018.