Patent claim segmentation through neural-based semantic analysis

By Danilo Silva de Carvalho

submitted to
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Written under the direction of
Associate Professor Minh-Le Nguyen

March, 2018

Patent claim segmentation through neural-based semantic analysis

By Danilo Silva de Carvalho (1520009)

A thesis submitted to
School of Information Science,
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Doctor of Information Science
Graduate Program in Information Science

Written under the direction of
Associate Professor Minh-Le Nguyen

and approved by
Associate Professor Nguyen Minh Le
Professor Satoshi Tojo
Professor Hiroyuki Iida
Associate Professor Kiyoaki Shirai
Associate Professor Yoshimasa Tsuruoka

December, 2017 (Submitted)

Abstract

The transition from the last century brought a considerable number of changes to international relations and trade, with a sharp acceleration of the Globalization process. The previous focus of industrial development is being changed from the production of physical assets to the concept of Intellectual Property, which is regulated in most countries by the patent system. With an increasing number of patents being applied and granted, management of innovation related information became an arduous task, leading to the development of a variety of approaches for its automation. While the use of Natural Language Processing techniques is well stablished among such approaches, the characteristics of this type of document still provides challenges to be addressed.

In this dissertation, I present a method for identifying and classifying textual segments from patent documents into relevant information types. It aims to facilitate the categorization and comparison of inventions by means of semantic analysis and is centered on claim sentences. The claims are the main information source in patent reviewing and litigation, so they are an important target for automation. The central aspects of the presented method, and major contributions of this work, are the annotation methodology for identifying elements of ideas in patent claims, that emulates the workflow of a patent professional, and the gathering and exploitation of multiple linguistic features from different sources. Those features enable the use of powerful Machine Learning techniques known as Deep Learning (i.e., Deep Artificial Neural Networks) even in cases where available training data is scarce. This research comprises the study of several aspects of patent information processing and describes the development of novel approaches to deal with them. In particular, the structuring of patent documents and claims from digitalized paper forms; the computational representation of lexical and semantic aspects of natural language; and the development and optimization of Deep Learning architectures for claim annotation. The analysis of claims is grounded on linguistic concepts and segmentation is done in a principled way, following patent expert advice. A manually annotated dataset for patent claims complete the set of contributions. Some basic premises explored in this work regard the syntactic constraints found in patent claims, which allow simplifications in appropriate methods or the use of techniques that would otherwise not be effective, while avoiding the otherwise effective solutions that fail when applied to claims. Experimental evaluation is performed for each one of the solutions presented, and the benefits and drawbacks of the developed methods discussed in detail. Results indicate that the automated segmentation of claims using the proposed method is viable and produces the desired results. The annotation principles guarantee the usefulness of the results to the patent expert community.

Keywords: Claim segmentation, Patent Information Processing, Semantic analysis, Deep Learning, Natural Language Processing

Acknowledgments

Firstly, I would like to thank my parents, who provided me good life examples and education, without which I would not have reached this point. While distant from my academic live nowadays (both mentally and physically), they shaped important life paths for me. Also to my relatives, especially my grandmothers, one for her moral, and many times financial support since my birth, and the other for always reminding me on how to go over hardships with a good sense of humor.

I would also like to thank my elementary school and high school teachers, who not only gave me the foundations for building the knowledge that I have attained today, but also motivated me to examine the many ways in which it can be shared with the world.

To my doctor and friend José Carlos Lino, without who I would not be alive to finish this work, and also for his inspiring work ethics and intellectually enriching conversations.

To my undergraduate school professors, João Carlos, Eber Schmitz, Mario Benevides, Geraldo Xexéo and Milton Ramirez, for motivating me into Artificial Intelligence research and opening my path to graduate school. Also to my external undergraduate supervisor André Freitas, for all the help and good advice so far, while trying to follow his footsteps as a researcher.

To my Master course supervisors prof. Felipe França and prof. Priscila Lima, for their special attention to me and big contribution to my development during this opening phase in my academic career.

My most sincere thanks to my supervisor prof. Nguyen Le Minh, for his trust in my capacities and for allowing me to conduct my work in an excellent environment. His advice and research critic were highly contributing to the results of this research.

To my laboratory friends, Takurou Ichino, Vu Duc Tran, Chien Xuan Tran, Long Hai Trieu, Tien Minh Nguyen and Viet-Anh Phan, my thanks for their support and camaraderie during the doctorate course.

I would also like to thank JAIST, its professors, administrators and staff, for providing the structure to learn and perform my research, as well as the social events and facilities that allowed me to greatly improve my understanding of the Japanese culture, language and customs.

Finally, I would like to thank CNPq and the Brazilian people for providing me the resources and the opportunity to study at JAIST, resulting in the development of this work and further advances in this research field.

Contents

Chapter 1

Introduction

The transition from the XX to XXI century, comprising the decades from 1990 to 2010, brought a considerable number of changes to international relations and trade, with a sharp acceleration of the Globalization process. A conjunction of political, economical and technological factors (e.g., the end of the Cold War, with consequent creation of new state economies, the expansion of telecommunications infrastructure), induced a strong movement towards international cooperation and the enactment of policies for facilitating trade and technology transfer in most of the world. Evidences of such movement can be found in the creation of national and international organizations during this period, such as the World Trade Organization (WTO).

One result of the new global arrangements is the ongoing transition of economic value from physical assets to *Intellectual Property*, as a late manifestation of what became known as the *Information Age*. Among the most valuable intellectual resources traded in the present day are the *patents*: government granted titles for exclusive commercial use of a specific invention, upon public disclosure of such invention. The value of a patent lies on the profitability prospects of its given invention and on its strategical position in delaying or changing actions of competitors, in a global market of fast-changing needs.

Recent intensification of Intellectual Property (IP) trade has given rise to a set of new businesses specialized in dealing with patents, including patent holding companies, IP analysts and consultancies. The prosperity of those enterprises brings incentives for more investment in IP, further accelerating the economic value transition from physical assets. Additionally, universities and other centers of academic research have also been brought under increasing pressure to produce Intellectual Property, with patent grants becoming part of many institutions' performance evaluation metrics. Incentives given to the involved agents have resulted in an explosion of patent application numbers, with China in particular almost tripling their application filings between 2009 and 2015 (Appendix A: Fig. A.1). This abrupt increase of patents makes managing information about innovation a difficult task, for which some of the aspects are approached in this work.

1.1 Motivation

Analyzing patent documents is done mostly with the goal of comparing their stated ideas to other ones available in the patent and academic literature. Comparing ideas is not a trivial task, involving expertise in both patent law and the technical domain of knowledge of the patents involved. People conducting patent analysis professionally are known as patent experts, and can be divided into two groups, according to their role in the patent system:

- *Patent attorneys*: Advise inventors on making their ideas patentable and represent them before the patent office. Alternatively, they help checking for patented ideas and avoiding infringement, or search for infringing competitors, in the case of companies.
- *Patent examiners*: The people working in a patent office, responsible for deciding if a patent should be granted or not, based on a set of openly defined rules. If granted, what ideas the patent should cover (scope).

For both groups, the task of comparing ideas in patents basically consists in narrowing down possible competing ideas in the relevant literature, so that innovation pertaining properties such as novelty and non-obviousness can be evaluated. These properties allow decisions to be made regarding patentability of an idea, or possible infringement of existing patents.

The challenge of analyzing patent information comes from its unusual language style and the wide range of possible relationships between ideas present in the literature. A patent document describes an invention in detail, and contains both concrete and abstract statements (e.g., physical specifications, process definitions, respectively). Such descriptions are laid out in complex sentences that are difficult to interpret even for humans, if not trained in the particular syntax style and vocabulary. Even for patent experts, with the aid of specialized document search systems, this can be a highly time-consuming task, which increases in cost with the number of patents and their complexity. A typical idea search comprises a thorough analysis of vast amounts of academic and legal documents.

Therefore, achieving a higher degree of automation in the categorization and comparison of patent information is an important goal. It lowers the cost of investigation for new inventions and patent disputes, and also could, in principle, shorten the time necessary for patent examination. Reductions in cost and time also mean a lower entry barrier for inventors with more limited resources, expanding socio-economical benefits and innovation avenues.

In order to facilitate this kind of automation, patent specialized methods for *Information Retrieval* (IR) and *Information Extraction* (IE) can be used. Their goal is to identify and obtain the most relevant parts of a document and organize them in searchable *knowledge bases* that are easily accessible for a wide array of computer algorithms. The process of collecting and structuring patent document information from natural language is collectively known as *Patent Information Processing*. Recently, this research topic has seen many advancements, in particular through the employment of *Natural Language Processing* (NLP) and *Machine Learning* (ML) techniques. However, while discoveries and improvements on NLP and ML methods benefit patent information processing directly, the complex content and interaction characteristics of those documents bring limitations to NLP and ML benefits, requiring more dedicated approaches.

Effectively processing patent documents thus requires a deeper understanding of their formation and working principles. This motivates a thorough analysis on the following aspects that guide this work:

Document accessibility:
How the relevant patent documents are made available by the different patent offices around the world: information types, formats and availability.

Document structure:
The way content is organized in a patent document: sections, metadata, textual and non-textual information, among other divisions.

Although patent documents tend to be long due to their detailed nature, the greater share of relevant (textual) information is concentrated in their claims section. The claims contain the definitive description of the objects of legal protection to be included in the patent and are the most important resource used in patent examination and litigation. For this reason, they are required to be written in an unambiguously and consistent way, thus being a better target for analysis.

Additionally, references, illustrations and other documents define relationships of important semantic value and should also be captured.

Linguistics:
The characteristics of the language used in the documents are fundamental for understanding their content and obtaining communication patterns that can be extrapolated for unseen input (i.e., new documents). Such characteristics can be divided according to their level of abstraction:

- *Lexical*: Which words are used and how they can be decomposed in their basic units.
- *Syntactic*: How sentences are composed (phrase order and relative positioning). Their grammatical structure.
- *Semantic*: The meaning of words and phrases in a given context.

Furthermore, the study and application of computational methods to a increasingly large amount of data also bring a set of efficiency and scalability considerations into attention. Possible solutions must also take into account the characteristics of both the data and methods developed to process it.

1.2 Patent Information Processing challenges

1.2.1 The patent system: premisses and processes

A *patent* is a government granted set of *exclusive rights* over the commercial use of an invention. The patent is assigned by a sovereign state to an inventor, or other physical or legal entity (the assignee) for a limited period of time, on the condition of disclosing an invention in detail. What constitutes an invention and how they should be disclosed depends on the country granting the patent, according to national laws and international agreements. However, a common understanding is that an invention is a product or process designed to solve a specific problem or to bring improvements to existing resources. The person or entity requesting a patent must describe carefully everything that comprises the invention, since the exclusive rights are granted only for what is written in the patent. The invention content detailed by the patent applicant is called *scope* of the patent. It delimits what is considered as part of the invention.

Patents are requested by those who want legal protection over the profiteering in their inventions. It is a way of ensuring the benefits obtained by the application of such inventions. To make commercial use of a patented invention, one should obtain permission from the patent holder, which usually charges a fee for granting such permission. Thus, patents enable revenue generation through creative work an incentivize capital investment in innovation. Typically, companies request patents as a way of protecting their technologies from competitors. Individuals and universities request patents to obtain a financial return for their efforts and also for recognition of their relevance to the society.

However, as the benefits from patent granting are important, the problems arising from the application of exclusivity rights are also of great relevance. An individual or organization that believes to have its patent infringed should prove that the alleged infringer is using its invention without permission. The alleged infringer will try to prove that the invention in question was not covered by the patent brought in the accusation, or alternatively try to prove that the patent in question is *not valid*. This means that companies should do extensive patent search in order check if the product they want to sell is not making use of some patented invention. Doing so avoids infringement and costly legal disputes. A similar kind of search can also be done by patent holders to find potential infringement from products currently on the market.

Depending on the number of patents in a given area of technology and the complexity of its inventions, the cost of searching and requesting patents can be high. This creates an advantage for larger organizations, which have more financial resources available. Those organizations can obtain a large number of patents and use them to limit innovation options to smaller competitors. Another problem is the granting of overly broadly defined patents, which may allow the inclusion of others' inventions in their scope. All the aforementioned problems must be resolved either during the grant process or a legal dispute over a patent. The latter option indicates a possible failure in the grant process, in which case a previously granted patent may be *invalidated* (i.e., rescinded).

The process of granting a patent starts with the filing of a *patent application document* by the inventor or assignee (the applicant) to the country's *patent office*. The application then enters *prosecution* phase, where it is reviewed by a *patent examiner* to determine if it meets the patentability requirements of the patent office. In case the application

fails any patentability criteria, objections are communicated to the applicant or their patent agent or attorney through an *Office action* (a type of document), to which the applicant may respond. Responses may include an amendment of the original description of the invention, adding details or removing parts that do not comply with the office's requirements.

After one or more office actions, a patent may finally be granted or rejected. If granted, a final patent document is issued after payment of issuance fees by the applicant. In some countries, the grant process enters an *opposition* phase before the actual issuance of the patent, where other interested parties may file an *opposition proceeding* to contest the validity of the patent to be granted. If the opposition is successful, the patent is then invalidated and cannot be issued anymore. Opposing a grant typically consists in showing *prior art*: proof that the invention was already known before the application filing. Alternatively, the opposition may show that the invention does not comply with some requirement from the patent office that was overlooked at first.

1.2.2 Patent Information Retrieval and Extraction: limitations and a proposed solution

Despite the existence of several patent information systems maintained by the patent offices and private companies, retrieving relevant information from patent documents is still a difficult task, for the following two major reasons:

- Semantics: Given the explanatory nature of the patent text, one could expect many ways of describing the same invention. This is further composed by the presence of both concrete and abstract elements in descriptions, as well as complex and sometimes new technical terms. Interpretation of such description elements are heavily dependent on context: the technological domain, purpose, detail level, and current document writing standards. This limits the usefulness of lexical-based approaches for search and pushes the boundaries of semantic representation in computer systems.

- Scope and amount of document relations: following the interpretation issue is the problem of determining the extent and manner in which a given pair of documents (and their parts) are related. Important relations include equivalence (synonymy), subsumption (i.e., a description "covers" another), entailment (i.e., a description "implies" another), among others. Those not only depend on the capacity of correctly representing the description semantics but also their connections. Furthermore, even considering only patent documents (i.e., excluding non-patent prior art), the number of possible relations between description elements can be very large, as it is not uncommon to find documents with hundreds of such elements, several per claim.

Those issues create a demand for patent analysis methods that are not only effective, but also efficient and able to scale well with the increasing amount of documents.

The processing of patent information is mostly done using techniques from the *Information Retrieval* (IR) and *Information Extraction* (IE) disciplines. The former is concerned with categorizing and selecting documents under a set of criteria (queries), while

the latter deals with identifying and obtaining specific information from inside individual documents. Modern IR and IE methods make feasible the work of patent experts by enabling the automatic construction of arbitrary field indexes, summaries and claim trees. Such resources became indispensable in order to narrow the amount of documents to manageable levels. However, tasks like technical classification, and prior art review are still done manually, due to the aforementioned semantics and description relationship challenges.

Additionally, while some improvements to *Natural Language Processing* (NLP) techniques like POS-tagging and Distributed Semantics benefit patent IR and IE methods, others are limited by certain characteristics of patent text. For example, constituency and dependency parsing methods are severely penalized by sentences that are very long, such as the ones found in claims. Unknown technical terms and spelling mistakes also tend to limit effectiveness of word embedding methods, as they typically provide a single way lexical mapping (word → vector).

However, as an alternative to retrieving and extracting information from the textual units defined by the patent document structure, it is possible to separate the ideas stated in a patent into several individual *elements*, which can then be analyzed and compared more precisely. This is a crucial part of a initial set of steps taken by patent experts when performing their work. Trying to emulate such methodology is the approach adopted in this work, which leads directly into the identification of ideas from the documents.

1.2.3 Claim segmentation

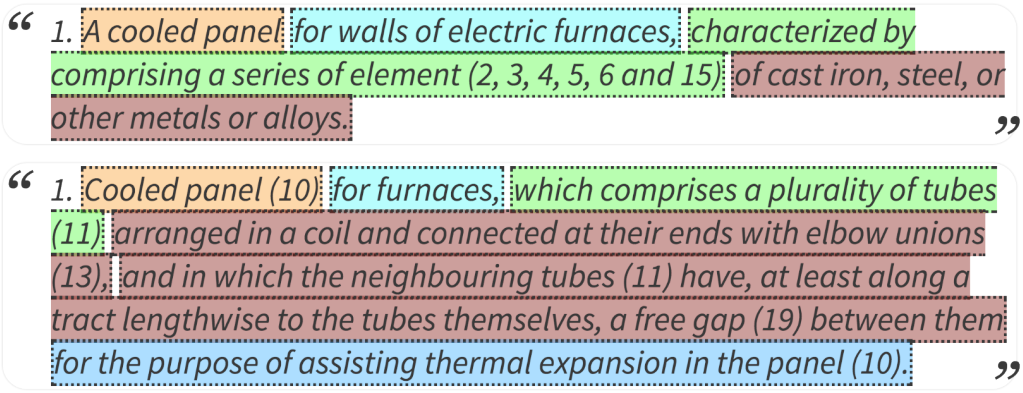
Identifying ideas in a patent implies to look into its claims: precise declarations of an invention's contents. Each claim is usually a single sentence describing a unique aspect of the stated invention, as can be seen in the following example:

> *“ 1. A cooled panel for walls of electric furnaces, characterized by comprising a series of element (2, 3, 4, 5, 6 and 15) made of cast iron, steel, or other metals or alloys ”*

A claim can be further decomposed into its descriptive elements: the core invention, function, materials and others. Elements that are not the core of the invention's idea are called *requirements* by patent experts.

> *“ 1. A cooled panel for walls of electric furnaces, characterized by comprising a series of element (2, 3, 4, 5, 6 and 15) of cast iron, steel, or other metals or alloys ”*

By categorizing such descriptive elements in this way, it is possible to compare different ideas regarding the corresponding elements, as can be seen in the following example:



Comparing such elements requires an appropriate representation for their meaning, as there are a variety of ways in phrasing the same content. Moreover, technical literature relies on composition of terms (e.g., “cooled panel”) to convey concepts, making computational representation more challenging.

Isolating such description elements is not a trivial task, as it is usual for claims to be very long and complex sentences, taking several minutes of a trained professional work time in some cases. An example of complex patent claim can be found in patent EP0915965(B1), on claim 13:

> “*A method of fabricating a bioreactor of any of claims 1 to 7, the method comprising: a) providing a hollow filament bioreactor cartridge, the cartridge comprising a housing containing a plurality of elongate hollow filaments each positioned within the housing substantially parallel to the central axis and defining an extrafilamentary space within the housing, each of the hollow filaments formed of a material which allows molecular transport therethrough, the housing further comprising a filament inlet port and a filament outlet port, said ports communicating through the hollow filaments to define a filament flow path, and a housing inlet port and a housing outlet port, said ports communicating through the extrafilamentary space to define an extrafilament flow path, the extrafilament flow path being isolated from the filament flow path such that a material in one path may enter the other path only by molecular transport through the material comprising the hollow filaments; and b) introducing a volume of a gellable material into the housing in a manner such that it becomes positioned in the region of the housing at which the housing outlet port is located or a short distance into the extrafilamentary space toward the housing inlet port, the volume of the gellable material such that, upon gelling, the resulting gel will form a hydrogel plug positioned in the extrafilament flow path to maintain a uniform flow across the extrafilament flow path.*”

A complete analysis of this claim can be found in Appendix C.

However, it is first necessary to access each individual claim from the document. Although the claims are all located on a specific section of the patent document, obtaining them may not be simple, depending on how the documents are made available by the patent office.

1.2.4 Patent document segmentation & Claim spotting

An important aspect of patent information processing is the one of *document accessibility*. Patent offices all around the world operate in different ways and this includes their method for receiving and publishing documents. Modern and resourceful ones like the EPO, USPTO, among others receive both paper and electronic applications and process them in digital systems from the beginning. This means that applications are recorded into a structured database that is also used for publication of patent information. In those systems, information extraction and retrieval are simplified, as the document fields (title, author, assignee, etc.) and sections (abstract, description, claims, etc.) are already accessible and often available for searches. Individual claims can be easily obtained in such cases.

On the other hand, there are patent offices that still receive paper applications exclusively, and publish their records as digitally scanned PDF forms. For those, a method for structuring the document information by dividing it into its sections is a fundamental step in analyzing its contents. The process of separating the individual sections of a document is a form of *patent document segmentation*. Identifying the individual claims from the rest of the document is called *claim spotting*.

Patent document segmentation and Claim spotting are usually preceded by basic document pre-processing tasks, such as *Optical Character Recognition* (OCR) to obtain the plain text from PDF forms and cleaning procedures to remove line markers, hyphenation and replace non-textual elements (e.g., tables and illustrations) with their appropriate representations (e.g., field contents and numbered captions).

This work presents a method for identifying and classifying description elements in patent claims, in the form of an automated annotation process for segmentation of the claim sentences. This method facilitates invention categorization and comparison by means of semantic analysis, as it attempts to emulate the initial stage of claim analysis performed by patent experts. This research comprises the study of several aspects of patent information processing and describes the development of novel approaches to deal with them. In particular, the structuring of patent documents and claims from digitalized paper forms; the computational representation of lexical and semantic aspects of natural language; and the development and optimization of Deep Learning architectures for claim annotation. The analysis of claims is grounded on linguistic concepts and segmentation is done in a principled way, following patent expert advice. A manually annotated dataset for patent claims complete the set of contributions. Experimental evaluation is performed for each one of the solutions presented, and the benefits and drawbacks of the developed methods discussed in detail. Results indicate that the automated segmentation of claims using the proposed method is viable and produces the desired results. The annotation principles guarantee the usefulness of the results to the patent expert community.

1.3 Organization of the dissertation

The remainder of this dissertation is organized as follows:

Patent processing is presented in a top-down manner, starting from document segmentation and claim spotting in Chapter 2, to claim segmentation in Chapter 3.

Chapter 2 explains the patent document structure and the solution developed for seg-

mentation, which includes claim spotting. The solution takes advantage of syntactic characteristics of this type of document, and a statistical method is used to obtain sentence representations with good discriminative power at low computational cost.

Chapter 3 explains the structural characteristics of patent claims and describes the different challenges involved in their analysis from both a linguistic and a patent examination perspective. Then, the process of automatic claim segmentation is detailed, starting from the basic problem formulation to the final Deep Learning architecture used to annotate the sentences. The application of each solution described in the previous chapters is presented as part of the complete mechanism.

Chapter 4 describes the experiments conducted on claim segmentation using the proposed method. The results are presented and discussed, followed by an analysis of error cases.

Finally, Chapter 5 concludes the dissertation with a summary of the work, its proposed and achieved goals, and a discussion on improvement avenues and future work.

Chapter 2

Document segmentation and claim spotting

One of the fundamental steps in analyzing patent documents is determining its basic structure, namely: abstract, description and claims. The information contained in each section can be used for different purposes, such as the use of abstracts for document summarization and classification. Separating those sections is a task known as *patent document segmentation*. Additionally, separation of the individual claims from the claim section can also be performed, and is called *claim spotting*.

The biggest patent offices (USPTO, EPO, JPO, among others) do all document processing electronically, meaning the documents are segmented from the start. However, there are several patent offices in which paper forms are still in use and are the only format of document publication. Those, when digitized through OCR [1], result in unstructured documents, for which automatic segmentation is needed before any further processing. The large amount of documents made available by the patent offices also compels the use of efficient methods, which can be executed without specialized equipment.

In this chapter, the approach developed in this work for patent document segmentation and claim spotting is discussed. In particular the use of the Term Order Probabilities (TOP) method for combining word embeddings into sentence representations at a low computational cost. Due to being only necessary for the increasingly uncommon cases of unstructured document publication, this is not a major contribution of this research. However, it is included in this work for its completeness.

2.1 Patent document structure

Patent relevant documents fall into the following types:

- Patent application: The original application document containing the description of the invention to be legally protected by exclusivity rights. Once received by the patent office, puts the patent in a "*pending*" state.
- Patent grant: The final patent accepted by the patent office and defining the protected invention. Puts the patent in a "*granted*" state, which should be followed

[1] Optical Character Recognition

by the issuance of the final patent document (similar to a property title), unless opposition invalidates it.

- Office action: A communication from the patent office to the applicant, changing the patent state to one of: "*provisional rejection*", "*final rejection*" or "*granted*", and detailing the reasons for the office's decision. May be followed by further office actions or a patent grant.

- Prior art: any document (e.g., academic paper, public presentation, etc.) communicating the whole or critical aspects of the invention contested, dating prior to the application filing date.

The office actions carry information about the changes made in a patent from its original form and is thus temporary in nature. The prior arts, on the other hand, are permanent but too wide in scope for structured searches. Therefore, this work focuses on patent application and grant documents, which comprise the two main states in a patent grant process: "*pending*" and "*granted*" and are permanent as well. To improve consistency in a knowledge base, only the grant documents are considered if they exist.

Both application and grant documents follow the same overall structure, which is composed of the following sections:

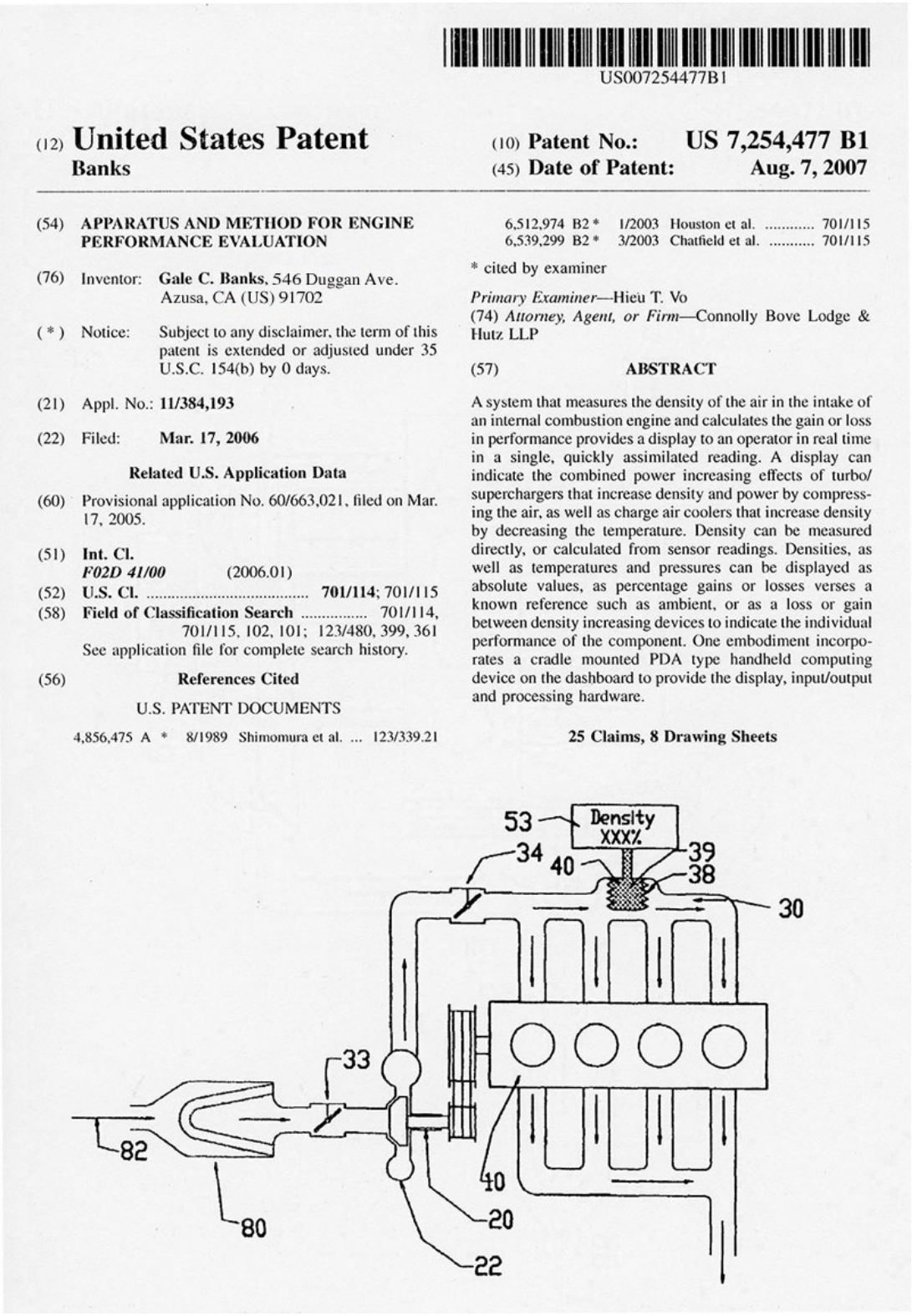
- *Preamble or header*: contains meta-information about the patent, such as inventor's name, application number and date, *IPC* classification (Section 3.3), among others.

- *Abstract*: A short summary of the invention contents described in the document.

- *Description*: A detailed description of the invention, history of creation, use scenarios, prior art and other information considered relevant by the applicant.

- *Claims*: A concise, but detailed description of the invention, that defines its scope unambiguously. Each claim is typically a single sentence that defines one aspect of the invention, making it unique. Most patent office communication and all opposition and litigation are centered in the claims.

- *Illustrations/Drawings*: Graphical representations of elements described in the invention. They are optional and depend on the type of invention.



US007254477B1

(12) **United States Patent**
Banks

(10) **Patent No.: US 7,254,477 B1**
(45) **Date of Patent: Aug. 7, 2007**

(54) **APPARATUS AND METHOD FOR ENGINE PERFORMANCE EVALUATION**

(76) Inventor: **Gale C. Banks**, 546 Duggan Ave. Azusa, CA (US) 91702

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **11/384,193**

(22) Filed: **Mar. 17, 2006**

Related U.S. Application Data

(60) Provisional application No. 60/663,021, filed on Mar. 17, 2005.

(51) **Int. Cl.**
F02D 41/00 (2006.01)

(52) **U.S. Cl.** **701/114**; 701/115

(58) **Field of Classification Search** 701/114, 701/115, 102, 101; 123/480, 399, 361
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,856,475 A * 8/1989 Shimomura et al. ... 123/339.21
6,512,974 B2 * 1/2003 Houston et al. 701/115
6,539,299 B2 * 3/2003 Chatfield et al. 701/115

* cited by examiner

Primary Examiner—Hieu T. Vo
(74) *Attorney, Agent, or Firm*—Connolly Bove Lodge & Hutz LLP

(57) **ABSTRACT**

A system that measures the density of the air in the intake of an internal combustion engine and calculates the gain or loss in performance provides a display to an operator in real time in a single, quickly assimilated reading. A display can indicate the combined power increasing effects of turbo/superchargers that increase density and power by compressing the air, as well as charge air coolers that increase density by decreasing the temperature. Density can be measured directly, or calculated from sensor readings. Densities, as well as temperatures and pressures can be displayed as absolute values, as percentage gains or losses verses a known reference such as ambient, or as a loss or gain between density increasing devices to indicate the individual performance of the component. One embodiment incorporates a cradle mounted PDA type handheld computing device on the dashboard to provide the display, input/output and processing hardware.

25 Claims, 8 Drawing Sheets

Figure 2.1: An example of a patent grant front page, from the United States Patent and Trademark Office (USPTO). Contains the header, abstract and an illustration.

2.2 Related work

Recent research on patent document segmentation for Information Extraction can be found in the works of Sheremetyeva [1], who presented a two level procedure for decomposition of section and claim structures grounded on deep linguistic analysis, and also Brugmann et. al. [2], who presented a complete document analysis system for patents, featuring several different techniques, ranging from document segmentation (section identification, claim spotting) to claim description analysis, entity recognition and document summarization. For comparison purposes, the segmentation method here described covers both the first level and third level's first stage of the segmentation hierarchy proposed by Brugmann et. al. [2], but using a Structured Perceptron, instead of CRFs combined with a set of heuristics. A direct attempt to structurally compare patent documents is presented by Huang et. al. [3], using Structured Self-Organizing Maps (SOM).

The distributional semantics approach for language representation (Section 2.5) also presents an attractive option for compact sentence representation, often through the composition of word embeddings [4, 5]. The most popular distributional representation approaches for sentences offer good performance on semantic relatedness and similarity tasks [4], but have a considerable computational cost compared to their word counterparts, which poses a problem for their application to big corpora, such as patent databases.

2.3 Patent document segmentation as a list membership problem

As a starting point, the segmentation task was defined as a membership problem, in which the elements are the document sentences. A sentence may only be part of a single section and the sections are sequences of sentences, so that one or two sentences in each one are boundaries. Thus, a tagging scheme including both membership and boundary information was used. Figs. 2.2 and 2.3 illustrate the tagging.

claims_B [I] [claim][:]
claims_M [1.] [A] [puppet] [adapted] [to] [be] [mounted] [on] ...
claims_M [2.] [The] [puppet] [of] [claim] [1], [wherein] [said] ...
3. ...
...
claims_E 11. The puppet of claim 10, wherein the neck ...

Figure 2.2: Example of sentence tagging for the claims section of a patent document. claims_(B/M/E) are the beginning, middle and end of the section respectively. [6]

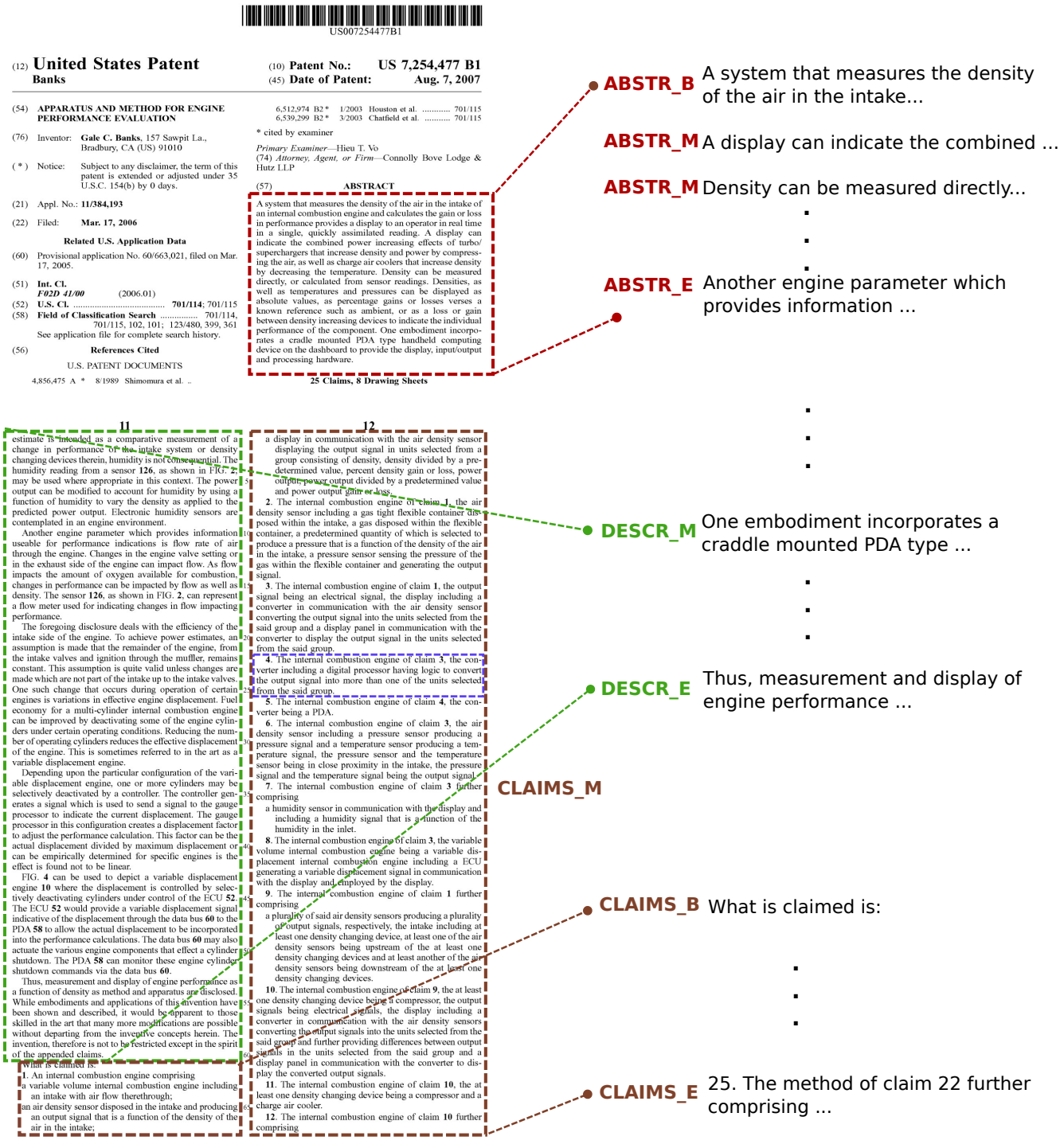


Figure 2.3: Example of a patent document segmentation.

2.4 The sentence representation issue

In order identify the correct section of a sentence, it is necessary to capture relevant information from it in the form of a single representation scheme. It is assumed that a semantic representation of a sentence's content would contain information relevant to its relative role in the document, thus helping to reveal its section.

Different methods for classification of sequence elements have distinct requirements for the representation scheme to be adopted. While a simple rule-based method for choosing the section and boundary markers might only need the original sequence of words for each sentence, other methods such as probabilistic graphical models: Hidden Markov Models (HMM), Conditional Random Fields (CRF) [7], among others, require engineering of features (e.g., sentence length, root verb tense) to be useful. Alternatively, Artificial Neural Network (ANN) models require sequence elements to be represented as vectors. Such vectors can be obtained in a variety of ways, some of them without direct human supervision.

Having in mind that the sentence representation should have good discriminative power, and aiming at obtaining representations from patents in multiple languages using a single approach, a choice was made to use *distributional-based* semantic representations.

2.5 Distributional semantics - text embeddings

Distributional approaches for language representation are also known as latent or statistical semantics. They are grounded in what is called the *distributional hypothesis* [8]. This concept stems from the notion that words are always used in a context, and it is the context that defines their meaning. Thus, the meaning of a term is concealed, i.e. latent, and can be revealed by looking at its context. In this sense, the meaning of a term can be defined to be a function of its neighboring term frequencies (co-occurrence). Using different definitions for "neighbor", e.g., adjacent words in *word2vec* [9, 10] and "modifiers in a dependency tree" [11], it is possible to produce a variety of vector spaces, called embeddings. Good embeddings enable the use of vector operations on words, such as comparison by cosine similarity. They also solve the data sparsity problem of large vocabularies, working as a dimensionality reduction method. There are, however, semantic elements that are not directly related to context, and thus are not well represented by distributional methods, e.g., the antonymy and hypernymy relations. Furthermore, polysemy can bring potential ambiguity problems in cases where the vectors are only indexed by surface form (an unidirectional mapping: word $\rightarrow$ embedding). Fig. 2.4 illustrates the process a basic *skipgram* method for obtaining word embeddings.

The reason text embeddings became so successful in many NLP tasks was not only due to their expressivity, but especially their practicality. They provide fixed size vectors that can be used as input for most Machine Learning methods, in particular *Artificial Neural Networks* which require this kind of input and are the basis for more powerful *Deep Learning* methods.

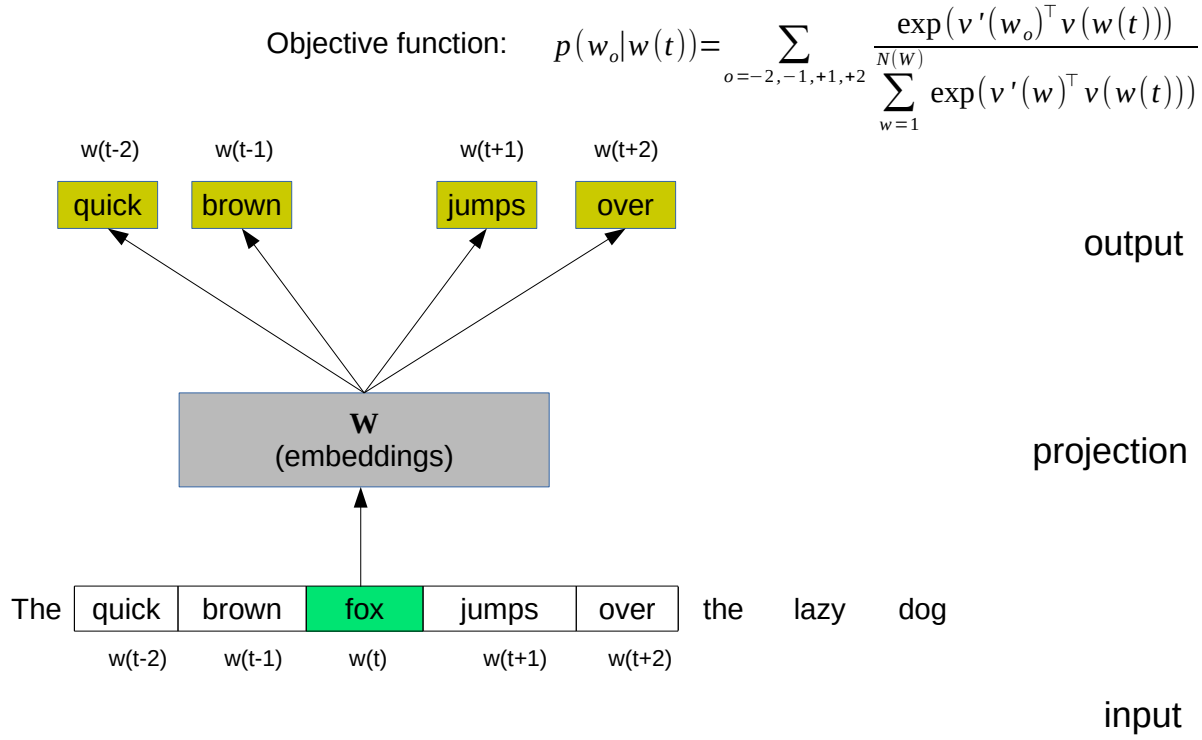


Figure 2.4: Simplified representation of the skipgram method: $p(w_o|w_p)$ is the joint probability of the output words $w_o = \{w(t-2), w(t-1), w(t+1), w(t+2)\}$ (the context), given the input word $w(t)$. $v'(w)$ is the projection vector for the word w in the vocabulary and $v(w)$ is the encoding vector (usually a unit or random vector) of word w. w_i is the word at index i in the vocabulary. The projection vectors compose the *projection matrix* W, which is adjusted to maximize the objective function.

2.6 Machine Learning application to sequence data: Artificial Neural Networks and Structured Learning

The topic of Machine Learning (ML) has been gaining interest in recent years due to its successful applications in diverse technological areas. From energy generation and distribution to medicine and commerce, ML is at the vanguard of solving practical problems that not long ago required specialized human input. With such accounts, ML is said to be bringing up a new era of automation, but it is not free of its own share of limitations and technical challenges.

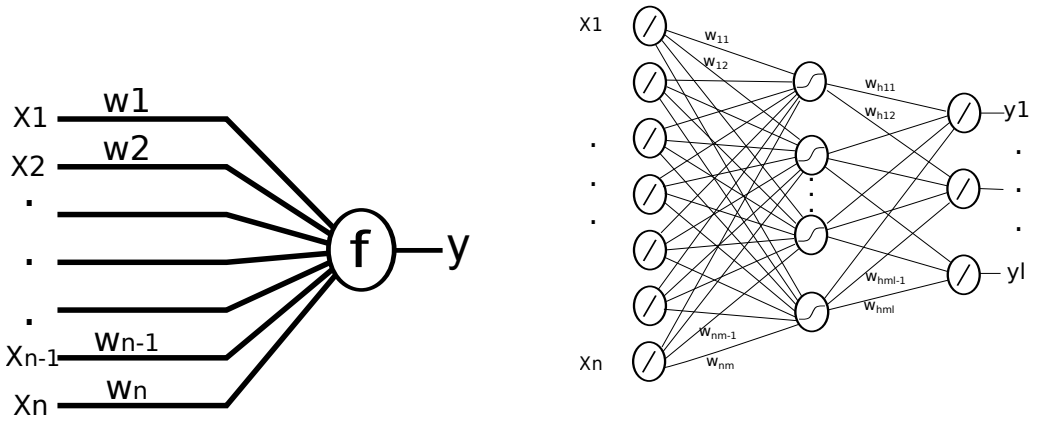
2.6.1 Artificial Neural Networks

In the years preceding the end of WWII, an important alternative to the algorithmic model of computation was presented by the work of McCulloch and Pitts [12]. In this work, they developed an electronic model that tried to imitate the connections and synapses found in the brain's neurons. The goal was to enable the solving of problems deem to be intractable at that time. The computational capabilities in this model were achieved

through properties that were inherent to a neuron network, which made trivial for humans some tasks that would be very complex, if not infeasible to program in a computer. The McCulloch and Pitts model achieved great prominence when put into practice in the beginning of the 1960's, by the work of Rosenblatt [13] in a system capable of image recognition, called *Perceptron*. The McCulloch and Pitts model and the Perceptron are considered the foundations of all Machine Learning models now known as Artificial Neural Networks (ANNs) [14, 15].

An ANN is composed of basic units called "neurons", connected to each other by multiple inputs and outputs called "synapses". Each synapsis has a "weight", responsible for modifying the output of a neuron, that will be taken as the input of another. Each neuron has an *activation function* (also called transfer function), that defines its output. Typically used activation functions are the logistic (sigmoid), hyperbolic tangent, and identity functions. The former two allow non-linear computation by the network. The weights can amplify or attenuate the output of a neuron and should be adjusted according to the desired function of the network. Adjusting the weights is a process called "*training*" and is done through a variety of algorithms, from which *backpropagation* is the most typically used [15].

The neurons in an ANN are organized in layers, starting from the one that receives the input signals (the *input layer*), which outputs are fed to a next set of neurons (*hidden layer*), so forth, until reaching the set of neurons that expose the network's output (the *output layer*). This basic network architecture is known as a *Perceptron* if it has no hidden layers or as *multi-layer Perceptron* (MLP), if it has one or more hidden layers. The number of layers is also known as the "depth" of the network. Fig. 2.5b illustrates the basic ANN model.



(a) Artificial neuron

(b) Multi-layer Perceptron

Figure 2.5: ANN model. Fig. (a) shows a neuron, the basic network unit, with inputs $X_1 \ldots X_n$, that are modified by the weights $w_1 \ldots w_n$. Each neuron has an activation function f that determines its output value y. Fig. (b) shows a multilayer Perceptron, being the first (input) and last (output) layers composed of linear activation (identity function) neurons, and the hidden layer composed of neurons with sigmoid activation function. Outputs $y_1 \ldots y_l$ are the attributes of the problem being solved by the network, e.g., the probabilities of a set of categories in a classifier.

Neural network models can also be viewed as simple mathematical models defining a function $f : X \rightarrow Y$, where each neuron function $f(x)$ is a composition of other functions $g_i(x)$, corresponding to the previous layer neurons. This can be expressed as $f(x) = K(\sum iw_i g_i(x))$, where K is the activation function, g_i are the preceding layer neurons and w_i are their corresponding weights. This in turn is commonly implemented in the computer as a chain of vector multiplications $f(x) = \hat{w}^\top \hat{g}$, where $\hat{w} = [w_1, ..., w_n]$ and $\hat{g} = [g_1(x), ..., g_n(x)]$.

2.6.2 Issues with unbounded sequence data & Structured learning

Among the problems handled by Machine Learning techniques, one particular type stands out due to both its inherent complexities and its ordinariness: the classification of unbounded (i.e., not fixed length) sequences of structured data. The essential issue with this type of data is that discrimination of one element in the sequence is not independent of the other elements. Depending on the structure type, dependency may manifest regarding adjacent elements or other arbitrary positions. Furthermore, the unbounded nature of such sequences means that absolute positioning should not be considered. A simple example can be found on the labeling of words in a sentence regarding their grammatical class, a task known as Part-of-Speech (POS) tagging. The label given to a word is conditionally dependent on both its neighboring words as well as their labels. Fig. 2.6 illustrates this notion.

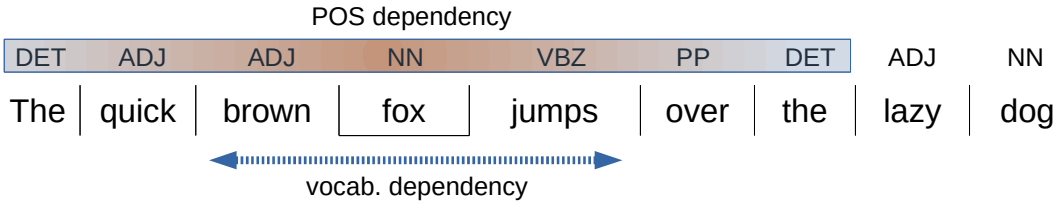


Figure 2.6: POS-tagging a sentence: the Part-of-Speech of a given word is conditionally dependent on both its neighboring words (vocabulary dependency) and on its neighboring labels (POS dependency). The degree of dependency diminishes with the distance from the target word.

To deal with this type of task, there is a class of Machine Learning techniques known as *structured learning* (or structured prediction). The representative examples of structured learning methods are the *Structured Perceptron* [16] and the *probabilistic graphical models*, in particular the *Hidden Markov Model* (HMM) and the *Conditional Random Fields* (CRF) [7]. Those methods attempt to predict labels for the entire sequence, instead of just one element at a time. This is done by maximizing the conditional probability of a particular label sequence given the observed elements.

The simplest method, the Structured Perceptron [16], is an extension of the traditional Perceptron algorithm in which a "joint feature function" $\Phi(x, y)$ takes as input both the original Perceptron input x and a candidate prediction y. The prediction is calculated with $\hat{y} = argmax(w^\top \Phi(x, y))$ and the weights $w^\top$ are updated using the incorrect answer

with the highest score y': $w \leftarrow w + \Phi(x, y') - \Phi(x, \hat{y})$. Its formulation then goes simply as:

Input: $D_{train} = \{(\mathbf{x}^1, \mathbf{y}^1) \ldots (\mathbf{x}^M, \mathbf{y}^M)\}$
set $\mathbf{w} = 0$
for $(\mathbf{x}, \mathbf{y}) \in D_{train}$ **do**
  predict $\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \in \mathcal{Y}^{\mathcal{N}}} \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}, \mathbf{y})$
  if $\hat{\mathbf{y}} \neq \mathbf{y}$ **then**
    update $\mathbf{w} = \mathbf{w} + \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) - \boldsymbol{\phi}(\mathbf{x}, \hat{\mathbf{y}})$
return w

where x is a sentence representation vector (the input), y is a unit vector representing a sentence tag category (<section>_<boundary-info>, the output), $\hat{y}$ is the predicted output from the model, w is the set of network weights and $\phi(x, y)$ is a joint feature function that maps the input and tag vector to a feature vector including the pairings (x, y) for the previous and next 2 elements in the sequence, as described by Collins [16], with null padding.

2.6.3 Using Structured Perceptron and distributional representations for membership and boundary identification

With a sentence representation scheme chosen, the next step was to choose a sequence classifier that could take such representations as inputs and predict the correct tags, specially the boundary ones.

While an attractive option from the computational cost aspect and good performance prospects due to the relatively low complexity of this task, a simple rule-based sequence classifier was discarded. This is because the number of small variations between documents across time would require a very large ruleset and posterior fine tuning of rules. Such variations are caused by changes in document standards across time, among other reasons. Furthermore, using a distributional-based representation approach allows more flexibility in obtaining models for multiple languages, but is not compatible with a rule-based classifier.

The use of distributional embeddings makes the inputs well suited for an Artificial Neural Network based approach. Considering the efficiency concern, the Structured Perceptron [16] was chosen over more complex models, such as Recurrent Neural Networks (RNN) (Section 3.6.1), due to its much lower computational cost.

To get the sentence vectors, the doc2vec [4] method was initially used, but it was taking a long time to train with the experimental corpus (see Section 2.8). This meant it would not be a viable approach for large-scale application, so an alternative approach was developed.

2.7 Term Order Probabilities: combining word embeddings for sentence representation

Considering the structural properties of syntactic constrained sentences, as typically found in patent documents, a method was developed to quickly obtain representative vectors of entire sentences. The goal was to speed up the segmentation process. It rests on the assumption that the higher structural regularity would decrease the amount of information lost due to the use of a less accurate method.

The method consists in calculating the probability $P(t_1, t_2)$ of any pair of terms (words, n-grams) t_1 and t_2 appearing in this specific order in the sentence. This value was called *Term Order Probability* (TOP) and can be easily calculated using the formula $P(t_1, t_2, d) = \frac{\#(t_1,t_2,d)}{\#(t_1,t_2,d)+\#(t_2,t_1,d)}$, where $\#(X)$ is the number of occurrences of X in the reference corpus and d is the maximum distance between t_1 and t_2. After calculating TOP for a corpus with n terms, the result is a matrix $\mathbf{P}^{n\times n}$ which is very sparse and can be efficiently stored and accessed. This information is useful for including basic structural information from a sentence in simple vector operations, as in the sentence embeddings described next. Fig. 2.7 illustrates the TOP matrix calculation.

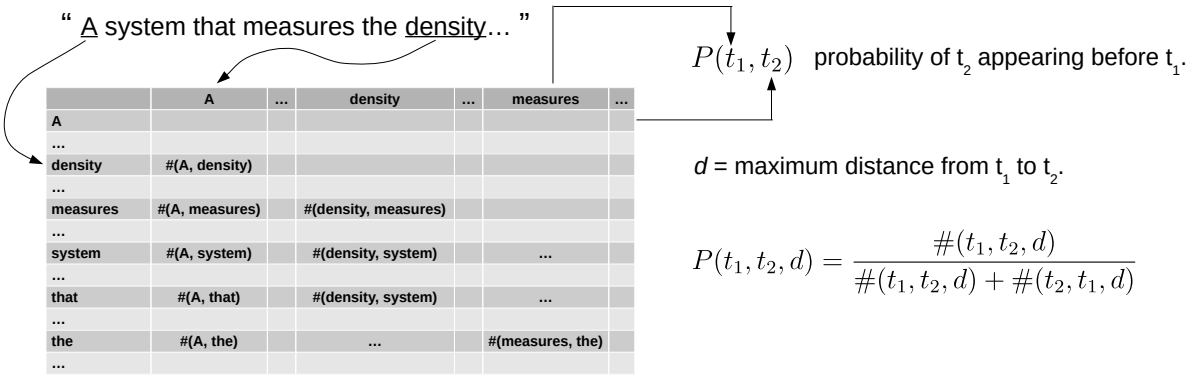


Figure 2.7: Term Order Probability matrix calculation. The $n \times n$ matrix (n = vocab. size) actually stores the frequencies of each occurrence of a specific word ordering, e.g. "density" appearing after "system". $P(t_1, t_2)$ is directly calculated from these frequencies.

To generate sentence embeddings using TOP, the following formula is used:

$$\mathbf{s} = \frac{\sum_{i=1}^{k} v(t_i) + \sum_{i,j=1:i<j}^{k} (v(t_i) + v(t_j)) * (1 - P(t_i, t_j))}{k + \sum_{i=1}^{k} k - i} \tag{2.1}$$

where t_i is a term, $v(t_i)$ is a term embedding and k is the length of the sentence. The resulting vector is the sum of the TOP-weighted combinations of all embedding pairs in the sentence. The range of j may be limited to create a fixed size window of distance d for each term, improving efficiency in longer sentences. The idea behind this formulation is that the contribution of each term to the sentence embedding is weighted by an "attention index" $(1 - P(t_i, t_j))$, representing how unlikely the term is to appear in that context. Thus, uncommon patterns have a higher contribution, helping to distinguish even between similar sentences.

This is in essence a modified bag-of-words approach, in which emphasis is put into word pairs that appear in uncommon positions in a sentence. In a corpus where sentences

in different sections have pronounced syntactic patterns, such as patent documents, this helps highlighting the section to which a sentence pertains.

While not as precise as Machine Learning-based sentence embedding methods, such as doc2vec [4], the cost of using TOP is much lower. The TOP matrix is calculated only a single time and can be built incrementally. The sentence embeddings obtained in this way are then used as inputs for the Structured Perceptron. Figure 2.8 illustrates the processing flow of a sentence in the document.

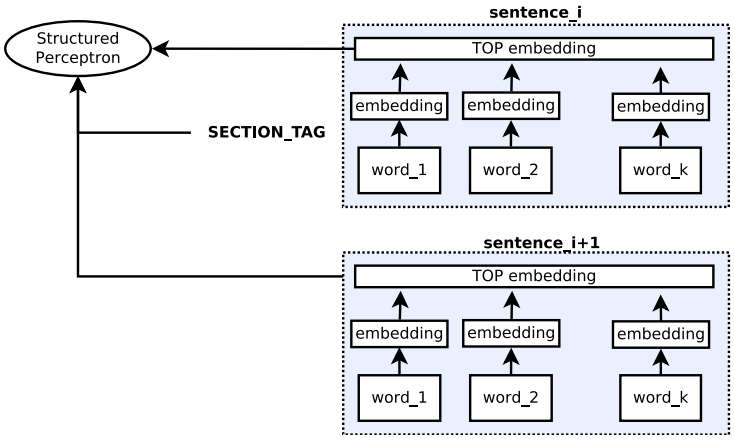


Figure 2.8: Sentence processing flow. Each TOP composed embedding and sentence label serve as input for the Structured Perceptron ANN. [6]

2.8 Experiments on patent text (USPTO)

Data acquisition and preparation

Experimental data was obtained from the United States Patent and Trademark Office (USPTO) patent repository, made available by Google [17]. USPTO documents were chosen due to their availability and the fact that all recent documents have annotations up to the level of claims, providing ground truth data for the tests. A set of 2000 patent documents from January to February 2015 (1K documents for each month) was used in the tests, totaling about 80 million sentences and 132 thousand terms, after n-gram modeling (N $\leq$ 3).

Experimental setup and methodology

The documents were split into sentences and each sentence was labeled according to the section and position it occupied, e.g.: CLAIM_(B/M/E) for the beginning, middle and end of the claims section, respectively. Optional sections, e.g., drawings, were not included. The training and prediction tasks were performed using the *Structured Perceptron* algorithm [16] in its *seqlearn* [18] implementation, taking the sentence embeddings of each document as input, with parameters: $decode = viterbi$, $lr = 0.1$, $iter = 10$ (implementation defaults). For the term embeddings, *word2vec* [9] was trained over the USPTO corpus for the set of documents from January to March 2015, with parameters: $d = 200$, $cbow = 1$, $window = 10$, $neg = 25$, $iter = 15$, $hs = 0$ and $sample = 1e - 5$. Sentence embeddings were generated in three different ways:

1. By using the formula described in Section 2.7.

2. By sum and average of the *word2vec* vectors.

3. By using the *doc2vec* implementation of the *Paragraph Vectors* [4] algorithm.

doc2vec parameters were set the same as *word2vec*, except $dm = 0$ and $iter = 5$ to make training time practical, so all embeddings have dimensionality = 200. *TOP* matrix calculation and doc2vec training were done over the same corpus section as word2vec. Accuracy was used as performance measure, calculated by taking the average ratio of correct predictions, normalized per class (tag), in the document collection. A document-wise 10-fold cross validation was performed 10 times, and the average results recorded, discarding the lowest and highest measurements. The term window size j for sentence embedding was adjusted between 2 and 8, running the cross validation once per value until finding the best. The sum and average approach (simplest) was used as a baseline. A running time measurement was also done, separated in training and test times. The test times represent the average CV fold time. TOP training time includes word2vec training time. The experiments were run on a Xeon 2GHz CPU (6 cores), 64GB of RAM computer. *Word2vec* and *doc2vec* training were done with 4 threads, all the rest being single-threaded.

Results & Discussion

Results from the experiments are presented in Table 2.1.

Table 2.1: Results from the document segmentation test. *sum&avg* means sum and averaging of all vectors in a sentence. *ws (window size)* is the maximum term lookahead applied to the sentence embedding formula in *TOP*.

Method	Period	Accuracy	Train time (min)	Test time (min)
Word2Vec sum&avg	Jan 2015	83.5%	116	0.2
Doc2Vec [4]	Jan 2015	67.3%	535	1.0
W2V TOP (ws = 4)	Jan 2015	**87.9%**	187	0.4
Word2Vec sum&avg	Feb 2015	78.0%	116	0.2
Doc2Vec [4]	Feb 2015	57.8%	535	1.8
W2V TOP (ws = 4)	Feb 2015	**84.5%**	187	0.6

The results indicate that the structural information included by *TOP* could improve the segmentation performance, while keeping a computational cost lower than the alternative tested. The obtained accuracy is adequate for real-world applications and is compatible with the findings of Brugmann et. al. [2], which reported a F1-score of 0.93 in this task for European patents, and Sheremetyeva [1] which reported a 100% accuracy result through supertag-based parsing, albeit with a much smaller document set (only 25 documents), however, a direct comparison could not be made regarding both works, due to the use of different sets of documents and public implementation not being available for both methods. The reduced performance of *doc2vec* can be explained by lack of training data, which is a dominant factor for that method. Its accuracy is expected to increase with a larger training set, but with training time also increasing as well, posing another advantage to *TOP*.

The results and output data from the patent document segmentation experiments can be accessed at `https://goo.gl/n4E1HJ`.

Error analysis

After analysis of the error cases, it was found that most prediction mistakes occur in the transition from the last description sentence in the abstract section to the beginning sentence of the description section. The description section to claims section transitions are less prone to errors due to the syntactic differences between those sections.

Some examples of misclassification can be seen in the following examples:

Sentence	Correct label	W2V+TOP predicted label
FIG. 1 is an upper perspective view of the cover for footwear showing my design.	DESCR_B	ABSTR_E
The invention claimed is:	CLSTT_S	DESCR_M
1. A method for manufacturing a heat-insulated pipe (1), comprising, ...	CLM_B	DESCR_M

When comparing the outputs from the word2vec with TOP tests with the averaged word2vec and doc2vec ones, the following observations could be done:

A) The averaged word2vec embeddings result in more intra-boundary misclassifications than word2vec + TOP. This indicates that the structural information included by TOP increases the discriminative power of the embeddings for this task, considering all other parameters equal. Boundary misclassifications are not significantly different between these two approaches.

Examples:

Sentence	Correct label	W2V+TOP	W2V sum&avg
More particularly, the present invention relates to sealing compositions which are particularly suitable for ...	DESCR_M	DESCR_M	DESCR_B
The prior-art rotary-piston machines with such planetary trains have the following common structural features;	DESCR_M	DESCR_M	DESCR_E
holding segments at the end of the forearms and in which the wrist joint further comprises two segments articulated in ...	CLM_M	CLM_M	CLM_E

B) The doc2vec embeddings result in a larger number of both boundary and intra-boundary misclassifications toward the dominant class (DESCR_M), indicating a low discriminative power for this task.

Examples:

Sentence	Correct label	W2V+TOP	Doc2Vec
The portions of the television and the graphical user interface shown in broken line form ...	DESCR_E	DESCR_E	DESCR_M
CLAIM	CLMSTT_S	CLMSTT_S	DESCR_M
The ornamental design for a television with graphical user interface, as shown and described.	CLM_S	CLM_S	DESCR_M

Sentence	Correct label	W2V+TOP	Doc2Vec
What is claimed is:	CLMSTT_S	CLMSTT_S	DESCR_M
1. A mobile recreational vehicle mounted sauna comprising:.	CLM_B	CLM_B	DESCR_M
2. The apparatus of claim 1, wherein said sauna is a bi-fuel sauna.	CLM_M	CLM_M	DESCR_M
3. The apparatus of claim 2, wherein said bi-fuel sauna may be heated by ...	CLM_M	CLM_M	DESCR_M

In all approaches, the boundary errors are likely to be reduced with by increasing the amount of training examples. The intra-boundary errors require improvements in the sentence representation scheme. Alternative distributional approaches and corresponding TOP adjustments may help in this regard.

Chapter 3

Claim segmentation

The purpose of analyzing patent claims is to be able to perform meaningful comparisons between elements of inventions described in different documents. Such ability allows the tracing of prior art and categorization of inventions, among other tasks dependent on detailed inspection of the claims' contents. Those tasks can be facilitated by obtaining semantic representations that can "approximate" the human interpretation of said elements, thus narrowing the scope where manual inspection is necessary.

The analysis can then be divided in two major steps: 1. Identifying and isolating the relevant terms associated with an element of the invention; and 2. Classifying the element according to its role in the invention, so it can be matched with its corresponding pairs. This process as a whole is called *claim segmentation*, as it results in a sequence of contiguous claim fragments that compose both a semantic and structural perception of the claim.

In this chapter, the principles behind the segmentation process are first explained, followed by a step-by-step description of the method developed in this work for the segmentation task, being its main contribution.

3.1 Claim structure and characteristics

Claims can be categorized into two types:

- Independent: declares the main matter of the invention, subject to exclusivity rights, indicating its basic characteristics. Example:

  "*1. An antimicrobial material comprising: at least one yarn comprising fine fiber of 1.0 denier or less ; and at least one yarn comprising antimicrobial fiber engaged with said yarn comprising fine fiber.*"

  In this claim, the main matter "An antimicrobial material" is basically described in what makes it unique.

- Dependent: details a previously declared matter, referring to one or more claims to be detailed. Example:

  "*2. The material of claim 1, wherein said material is woven.*"

  In this claim, the material previously declared is further detailed. A reference claim number is used for indicating to which claim(s) the details apply.

The dependency relationships between the claims can be represented as a directed graph, often called "*claims tree*", as there should be no cycles in a well written document. The root of a claim tree is an independent claim, which means a document that has more than one independent claim will have a *claims forest*, i.e., a set of claims trees. The claims tree is a very important structure in understanding the content of a patent and is often referred by patent experts in their analysis and communications. It facilitates the tracking of changes to the patent, especially during the review phase, when claims may be added or removed to satisfy patentability requirements. Fig. 3.1 illustrates the patent claims trees from patent WO0074470 (claims in Appendix B).

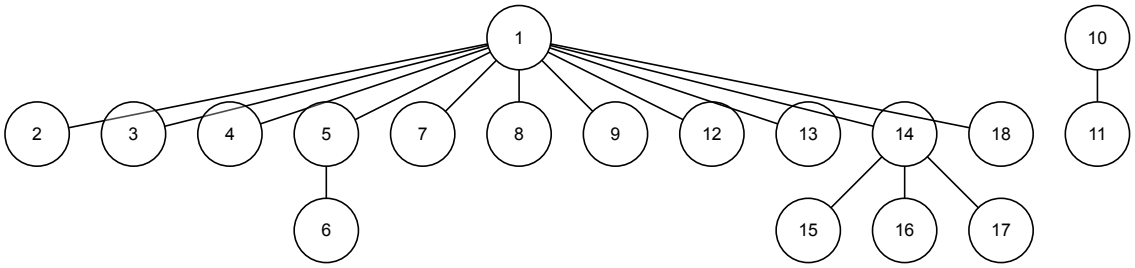


Figure 3.1: Claims trees from patent application WO0074470 (Appendix B). There are two independent claims (1 and 10) from a total of 18. The lines represent the dependency relationship between claims, with the children nodes being dependent on the parent ones.

3.2 Related work

The work on information extraction from patent documents is characterized by the difficulty in isolating domain knowledge from the text, given the vast domain coverage of patents. Therefore, it is often associated with upper and domain ontology research, both in the form of taxonomic information extraction or entity recognition, and in the form of ontology construction from term and relation extraction. Semantic analysis is usually employed for the latter form, but benefits greatly from ontology matching, which is another aim of the research.

Important work on patent information extraction can be found in Ghoula et al. [19], which described a method for generating semantic annotations on patent texts, using the document structure and a multilevel ontology annotation scheme, supported by a combination of NLP techniques. Although this approach is fast and well aligned with a web semantic perspective, it depends on structured documents and an existing domain ontology for extraction of information inside the patent claims. Taduri et al. [20] proposes a patent system ontology, aiming to standardize the representation from different information sources, initially focusing on US patents office and court records. Yang and Soo [21] presented a method for extracting conceptual graphs from claims using syntactic information and a background ontology, also focusing on US patents claim structure.

Hung et al. [22] proposed a method for dividing legal text sentences into previously defined logical structures for Japanese $\leftrightarrow$ English translation purposes, using a statistical method. The presented segmentation model is an application of semantic function modeling. Related work on semantic analysis include the method for part-whole domain inde-

pendent relation extraction, presented by Girju et al. [23], and the weakly-supervised algorithm for generic pattern relation extraction presented by Pantel and Pennacchiotti [24], both using semantic annotation learning. Nguyen and Shimazu [25] presented a method for semi-supervised semantic parsing, which applies a form of semantic function extraction, in the form of meaning representations. A generative semantic alignment model is used to map sentences to meaning representations for training.

Recent work in typification of patent information directly from text is presented by Roh et. al. [26], in which a technology information typology is used to categorize and cluster keywords into "layers" of technical markers. The patent document is progressively structured through such layers.

This research differs from related work in that it does not tackle information extraction directly, but rather presents a pre-processing step that facilitates any further information extraction approach. It also differs in the way it deals with generalization issues, using semantic annotation for patent claims as in Ghoula et al. [19], but with a ontology independent approach, and also by building a multi-faceted language representation which is strongly grounded on linguistic concepts. Such representation enables efficient textual pattern recognition in both structural (morphology, syntax) and semantic levels. Machine Learning plays a fundamental role in the construction of this representation, but is strongly supported by extensive analysis and the development of automatic structuring methods for textual corpora. In this context, Deep Learning techniques serve as a powerful tool for information representation. In particular, the long chains of internal dependencies (requirements) typically found in patent claims, are well suited for the application of LSTM recurrent neural networks, due to their sequence mapping capabilities, as shown by Sutskever, Vinyals and Le [27].

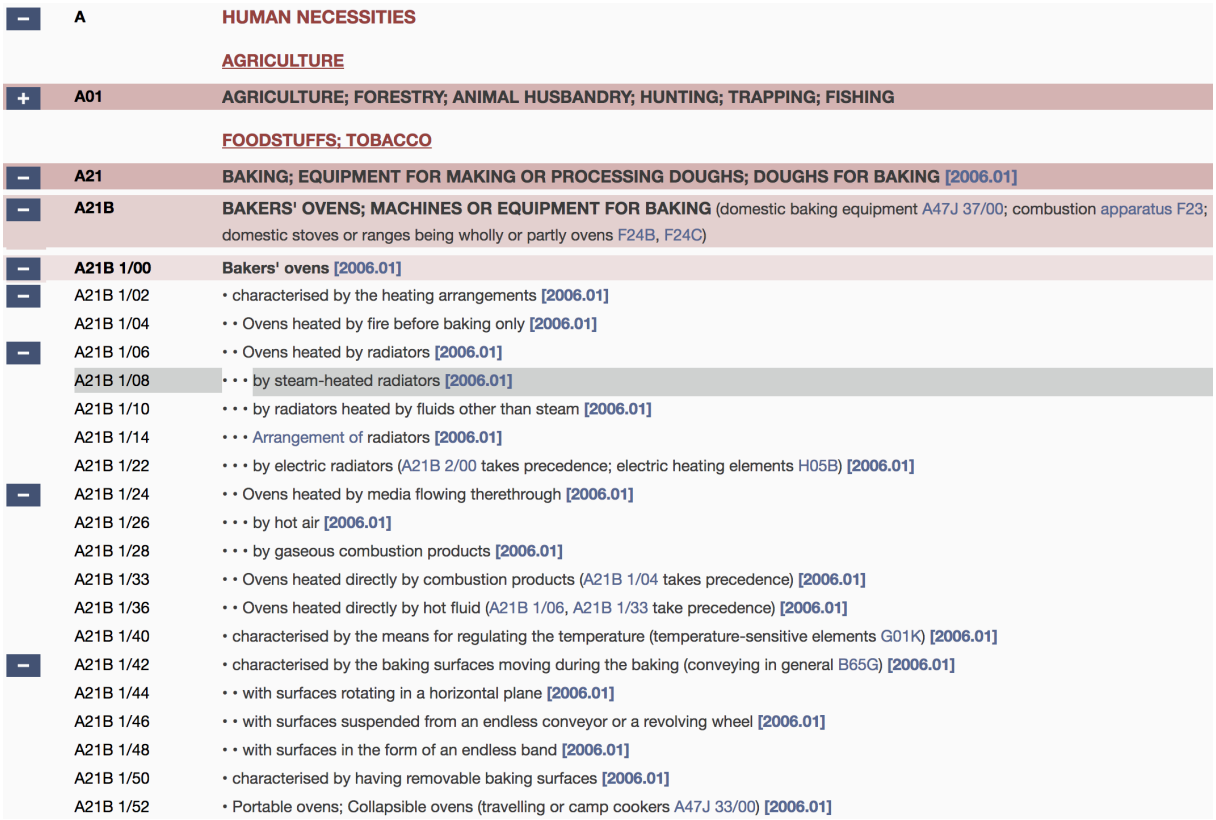
Despite the benefits perceived in such semantic decomposition of claims in description elements, this author could not find other research on this particular task at the present.

3.3 The International Patent Classification (IPC) system

The International Patent Classification (IPC) is a hierarchical classification system established by the Strasbourg Agreement 1971, which is administered by the *World Intellectual Property Organization* (WIPO). It provides a set of language independent symbols for classifying patents and utility models according to the different areas of technology to which they pertain [28]. The IPC is used in the adopting countries to classify the content of patents in a uniform manner, facilitating the retrieval of documents and making feasible the search for *prior art* across different patent offices. The system is updated annually by a committee of experts from the member states and from patent organizations such as the *European Patent Office* (EPO).

The IPC system comprises a set of eight core *sections* (A - H) that are divided in topic groupings called subsections. The subsections are then comprised of *classes* indicated by a two-digit number. Each class comprises one or more subclasses, indicated by a single capital letter. The subclasses are divided into groups, which are indicated by a number and a subgroup code. A subgroup is indicated by a decimal-like notation for the purpose

of ordering, and a hierarchical "dot" notation for determining further subdivisions. Each IPC symbol has a title that describes the technological contents of its grouping. Fig. 3.2 shows a fragment of the IPC hierarchy starting from the section **A** (Human Necessities) up to the subgroup level.



−	A	HUMAN NECESSITIES
		AGRICULTURE
+	A01	AGRICULTURE; FORESTRY; ANIMAL HUSBANDRY; HUNTING; TRAPPING; FISHING
		FOODSTUFFS; TOBACCO
−	A21	BAKING; EQUIPMENT FOR MAKING OR PROCESSING DOUGHS; DOUGHS FOR BAKING [2006.01]
−	A21B	BAKERS' OVENS; MACHINES OR EQUIPMENT FOR BAKING (domestic baking equipment A47J 37/00; combustion apparatus F23; domestic stoves or ranges being wholly or partly ovens F24B, F24C)
−	A21B 1/00	Bakers' ovens [2006.01]
−	A21B 1/02	• characterised by the heating arrangements [2006.01]
	A21B 1/04	• • Ovens heated by fire before baking only [2006.01]
−	A21B 1/06	• • Ovens heated by radiators [2006.01]
	A21B 1/08	• • • by steam-heated radiators [2006.01]
	A21B 1/10	• • • by radiators heated by fluids other than steam [2006.01]
	A21B 1/14	• • • Arrangement of radiators [2006.01]
	A21B 1/22	• • • by electric radiators (A21B 2/00 takes precedence; electric heating elements H05B) [2006.01]
−	A21B 1/24	• • Ovens heated by media flowing therethrough [2006.01]
	A21B 1/26	• • • by hot air [2006.01]
	A21B 1/28	• • • by gaseous combustion products [2006.01]
	A21B 1/33	• • Ovens heated directly by combustion products (A21B 1/04 takes precedence) [2006.01]
	A21B 1/36	• • Ovens heated directly by hot fluid (A21B 1/06, A21B 1/33 take precedence) [2006.01]
	A21B 1/40	• characterised by the means for regulating the temperature (temperature-sensitive elements G01K) [2006.01]
−	A21B 1/42	• characterised by the baking surfaces moving during the baking (conveying in general B65G) [2006.01]
	A21B 1/44	• • with surfaces rotating in a horizontal plane [2006.01]
	A21B 1/46	• • with surfaces suspended from an endless conveyor or a revolving wheel [2006.01]
	A21B 1/48	• • with surfaces in the form of an endless band [2006.01]
	A21B 1/50	• characterised by having removable baking surfaces [2006.01]
	A21B 1/52	• Portable ovens; Collapsible ovens (travelling or camp cookers A47J 33/00) [2006.01]

Figure 3.2: Fragment from the IPC hierarchy showing some of the first symbols of section **A**: Human Necessities, up to the subgroup level.

In Fig. 3.2, examples of IPC symbols are:

- **A** (section): Human Necessities.
- (subsection): FOODSTUFFS; TOBACCO
- **A21** (class): BAKING; EQUIPMENT FOR MAKING OR PROCESSING DOUGHS; DOUGHS FOR BAKING.
- **A21B** (subclass): BAKERS' OVENS; MACHINES OR EQUIPMENT FOR BAKING.
- **A21B 1/00** (group): Bakers' ovens.
- **A21B 1/02** (subgroup, level one): characterized by the heating arrangements.
- **A21B 1/06** (subgroup, level two): Ovens heated by radiators.
- **A21B 1/08** (subgroup, level three): by steam-heated radiators.

Each sub-level in the IPC hierarchy represents a higher specialization of the technology in the parent level. A patent document can receive one or more IPC symbols, as deemed appropriate during the review process. There is an approximate total of 70000 symbols in the IPC system, allowing fine-grained searches in a universe that grows by millions of documents per year.

3.4 Claim element analysis: a patent examination perspective

As an initial step in their process of patent analysis, patent experts will typically break up the claims in a specific manner, considering different ways of categorizing the elements in the claim's stated idea. Two categorization aspects are usually covered:

- *Structural*: the way in which the claim elements are organized, forming a hierarchy of detail.
- *Technical*: the technical qualities of the element's contents related to each other and to the main matter of the invention.

In a structural view, the claim elements are viewed as part of a taxonomy, which has the invention main matter at its top. An element can be either the main matter or a *requirement* to another element. The interpretation of an element can only be complete when including all its requirements. Fig. 3.3 illustrates the structural view of a claim.

In a technical view, an element can either describe a component or a function (or purpose) of the main invention matter or another element. The interpretation of each element is independent, and the relationships between elements define the interpretation of the invention. Interpretation of both elements and relationships is done in a technical basis, by an expert of the given invention's field. Fig. 3.4

A patent expert will start a claim analysis by doing structural segmentation, identifying the claim's main invention matter and its immediate requirements, following requirement paths in depth-first or breadth-first manner, according to personal preference. Technical segmentation is optionally done after that, depending on the needs of the patent applicant or patent office [1].

Furthermore, the levels in a structural view are related with the International Patent Classification system hierarchy, being such level alignment a critical point in the classification of a claim according to the IPC system.

To accommodate the representation and classification of claim description elements, the notion of *semantic function* is here defined:

A *semantic function* is an logic abstraction for a meaning unit in text, which can be expressed in one or more words. It is represented as a predicate $F(X, ...)$, where F is the function and X is a term participating in the meaning unit denoted by the function. Some simple examples are $Thing$(the dog) or $Cause(flood, rain)$. They can be understood as a generalization of the *semantic role* notion, as F is not limited to a thematic nature and accepts an arbitrary number of participating words, so it can cover phrases, sentences or documents.

[1] Professional patent analysis counseling was provided by JAIST patent attorney Ms. Shoko Mitani, under a university contract.

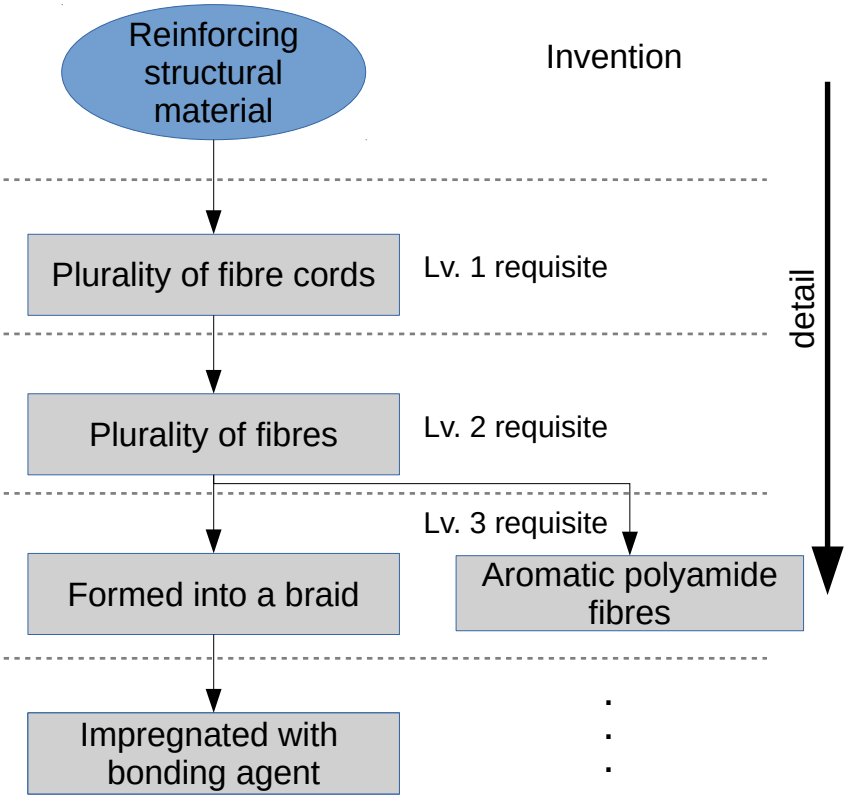


Figure 3.3: Representation of the structural view of patent claim fragments from patent EP-0206591-B1 (EPO). Each level describes a different degree of detail. The interpretation of an element requires the inclusion of all its requirements. The invention is thus described through all the claims elements.

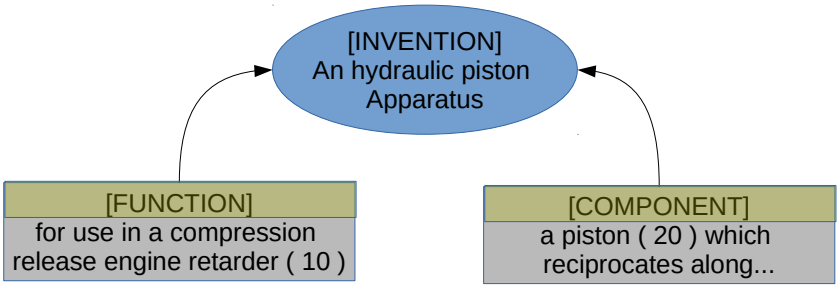


Figure 3.4: Representation of the technical view of patent claim fragments from patent EP-0552596-B1 (EPO).

The meaning represented by a semantic function is always bound to a specific context and depends on it to manifest itself, e.g, the concept of "protein" in a biochemistry book, for which there are thousands of instances, or the concepts of requisite and effectuation in legal text. Therefore, a set of semantic functions represent a specific point of view (i.e., a filter) about any given text. The document section categories shown in Fig. 2.3 are an example of semantic function set, in the context of patent document organization.

With that in mind, the first semantic functions to be defined in the context of claim

analysis are:

- *INVENTION*: The main invention matter. Object of legal protection.
- *REQT[#Lv.]*: A requirement, specifying the level in the structural view of the claim.
- *CLAIM_NUM*: The sequential number of the claim. Serves as identifier.
- *CLAIM_REF*: A reference to another claim in the patent. Determines the claim as *dependent* (see Section 2.1).
- *CLAIM_REF_NUM*: The parent claim identifier in a claim dependency.
- *FIGURE_REF*: An identifier for an illustration in the document.

Additionally, the following technical view semantic functions were defined for posterior usage.

- *DETAIL_START/END*: Marks the start and end of the invention matter detailing.
- *COMPOSITION*: Describe a set of components of the referring element.
- *FUNCTION*: Describes a function or purpose of the referring element.

3.5 Annotation principles and the creation of a reference corpus

In order to facilitate the creation of a reference corpus of claim segment annotations, a set of annotation principles where established, based on professional practices of patent experts [2]. This would allow providing good quality claim segmentation examples for training Machine Learning methods.

The annotations intend to imitate a process of "marking up" the text at different levels of detail. So it is possible to start with broader "sections" (e.g., invention, requirement) and then proceed to more detailed marks (e.g, claim references). The principles can be summarized as follows:

- Annotations are done left to right.
- Annotations are hierarchical. In other words, the overlap between marks is total or none. There is no partial overlap between annotations.
- Only the defined semantic functions are valid and permitted in the annotations.
- Requirement levels are counted towards the invention declared at each claim.
- Detailing a claim element increases the requirement level by one, creating a requirement "branch".

[2]Professional patent analysis counseling was provided by JAIST patent attorney Ms. Shoko Mitani, under a university contract.

- Requirement levels may be nested for clarity (cue) purposes, but this is optional.
- Claim references are regarded as a single requirement level. The remaining requirements of a dependent claim start from one level past the claim reference.

Fig. 3.5 illustrates a pair of annotated claims.

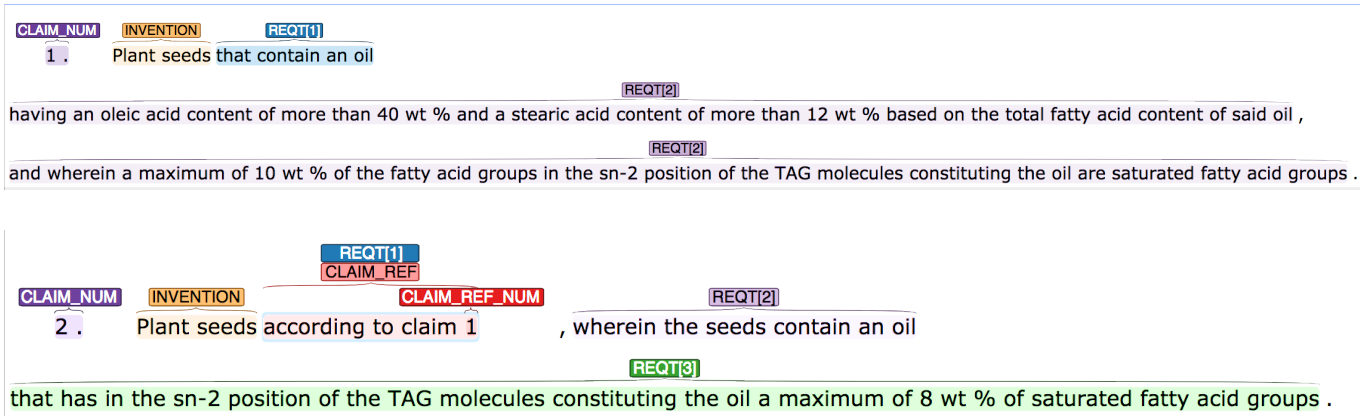


Figure 3.5: Annotated claims from patent WO-0074470-A1 (WIPO). In the first claim, there are two requirement branches of level 2, detailing the first level requirement "that contain an oil" of the invention. The second claim contains a reference element "according to claim 1" that defines it as *dependent* on the referred claim.

To start the reference corpus, a set of 6490 patent documents was randomly sampled out of the EPO/WIPO public patent database, including applications and grants. The sampling was done so that it selected 10 patents from each of the 649 subclass level symbols of the IPC. From this set, 20 documents were randomly selected for manual annotation, covering the following major topics: Agriculture, Biochemistry, Building, Computing, Communication, Electric Power, Inorganic Chemistry, Machines, Medicine and Vehicles.

3.6 Machine Learning application to sequence data: Deep Learning

Despite allowing representation of complex functions, the original Perceptron was limited to the class of *linearly separaple* problems, failing to model otherwise simple non-linear functions such as the "exclusive or" (XOR). Once this limitation was brought into attention [29] in the late 1960's, interest in the ANNs and in the Artificial Intelligence field in general waned. This led to the later development of alternative ML methods, such as the Support Vector Machines (SVM) [30], which solved a broader class of problems.

ANNs would see a strong revival in the early 2000's, when optimizations of learning algorithms and the improvement of computer hardware made the use of networks with several hidden layers feasible. Those networks, now called *Deep Neural Networks* (DNNs), could model highly complex functions due to their increased depth, having an arbitrary number of parameters. Consequently, they found applications in Computer Vision and other difficult classification tasks. This led to further research on other types of ANN architectures that were also "deep", such as the Boltzmann machine (BM) and Recurrent

Neural Network (RNN). Those methods are collectively known as *Deep Learning* (DL). In the late 00's, Deep Learning got a strong impulse from the advancements in *General Purpose Graphical Processing Unit* (GPGPU) hardware [31], which was followed and the development of specialized programming libraries for DL systems development. Speed gains with GPGPUs were at least of an order of magnitude when compared with the use of CPUs, allowing the use of much bigger data sets and quicker experimental cycles. From this point, DL research intensified and started showing state-of-the-art results in Natural Language Processing, among other fields.

There are, however, some points of attention that greatly affect the outcome of research when developing DL systems:

1. Amount of data shall be proportional to the complexity of the network.
The deeper a ANN is, the larger the number of parameters (weights) it will need to adjust in order to be expressive, i.e., to allow representation of more complex functions. Therefore, training complex networks will often require large amounts of observations, which may not be easily obtained (e.g., due to cost or physical limits). The amount of data may also bring high computational cost, even considering the use of GPGPUs.

2. Network type and topology.
A Deep Learning system is classified according to the type of task it will perform and how its neurons are organized. Knowing both is a essential step in deciding which model to use and the appropriate procedure to train it.

- Machine Learning task type:
  - *Supervised*: The system learns from example outputs (labels) given for each observation in training data, so it can predict correct outputs for new (unlabeled) data.
  - *Unsupervised*: The system learns regularities, patterns and structures from the training data, so it can find matches on unseen data or generate new data according to the learned structures.
- Network topology:
  - *Feedforward*: The network can be viewed as a *Directed Acyclic Graph* with sources at the input neurons and sinks at the output neurons. This is the most traditional ANN topology.
  - *Recurrent*: The network has a cycle, allowing arbitrary sequences of inputs to be processed by changing their internal memory state (see Section 3.6.1).

3.6.1 Deep Learning for structured data: recursive & recurrent networks

The formulation for the Structured Perceptron (Section 2.6.2) is adequate to predict sequence data, but lacks the representation power of Deep Neural Networks. Therefore, a sequence capable Deep Learning solution would be required to model complex dependency functions between sequence elements.

Considering the fact that a Deep Neural Network can have an arbitrary number of layers, a simple solution to the problem of modeling complex structures, including sentences, would be to ascribe a layer for each element in the structure. This arrangement could then be trained by adjusting the weights in each layer according to the outputs given by it, should it be the final hidden layer in the network. However, this solution is not only limited to fixed size structures, but requires an amount of resources proportional to the size of the structure.

Those size limitations can be avoided by having one or more hidden layers to represent **any** element in the structure. This is done by changing the network architecture to allow cycles, feeding the previous state of one or more layers into a posterior observation. Fig. 3.6 illustrates this concept.

The way in which the previous state of the network is combined into the current state is what differentiates such ANN architectures. If the layers ascribed to more than one element are combined into a single one, it is called a *Recursive Neural Network* (RvNN). If only one element's layers are combined from the previous network state into the current one, then it is called a *Recurrent Neural Network* (RNN). While RNNs can be used to model functions over sequences, RvNNs can also be used for tree structures.

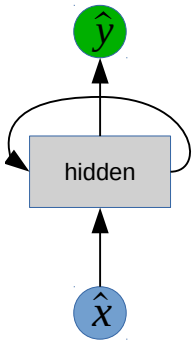


Figure 3.6: A simplified illustration of an Recurrent/Recursive NN.

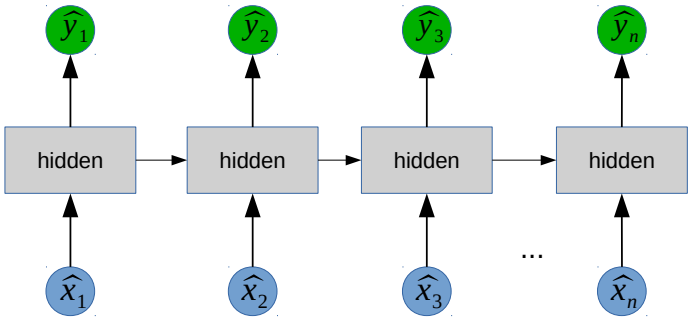


Figure 3.7: The representation of an "unrolled" RNN. Each time step maps an input vector $\hat{x}$ into an output vector $\hat{y}$.

A RNN can be seen as a finite list of time steps in which each step maps an input vector $\hat{x}$ into an output vector $\hat{y}$. This is known as the "unrolled" form of the network (Fig. 3.7). This understanding of the RNN architecture makes evident two of its main drawbacks, when compared to a probabilistic graphical model: (a) it only takes into account what was in the previous steps of the sequence, and (b) elements long past the current time step will suffer from "forgetfulness" of the network. The latter happens because any adjustments made to their hidden layer weights will have to go through all the time steps from the current one, being heavily scaled by the successive vector multiplications (i.e., the vanishing gradient problem).

A common strategy to solve (a) is to use a second set of hidden layers and use them to process the sequence in backward order. The output of the forward and backward layers is then combined, e.g., through mean or concatenation. This is called a *Bi-directional RNN* (Fig. 3.8).

To solve (b), a different formulation for the learning rule of the hidden layers was developed, combining different activation functions and a memory element to create the *Long Short Term Memory* (LSTM) RNN module [32]. LSTM mitigates the vanishing

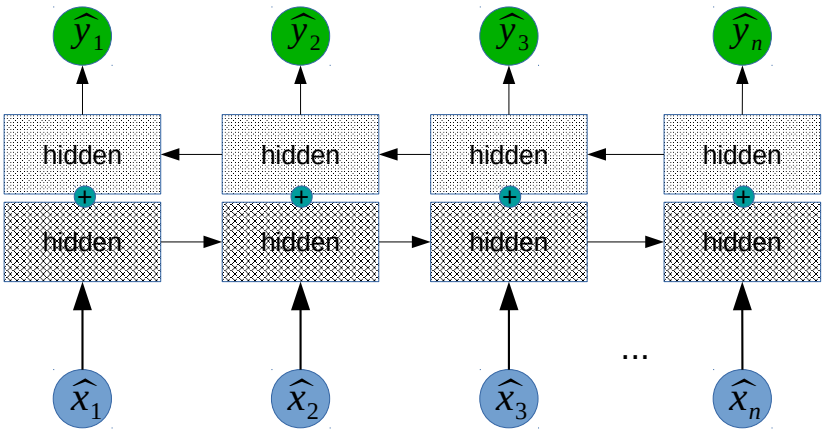


Figure 3.8: The representation of an bi-directional RNN. The (+) symbol means that the output of layers in both directions is concatenated at each time step, so the output dimension is doubled.

gradient problem, being appropriate for both short and long sequences alike. For this reason, most RNN implementations currently use LSTM modules instead of "plain" hidden layers, since they can be used in any case a plain RNN would be. RNNs using LSTM modules are called LSTM networks (LSTMs).

The use of probabilistic graphical models (HMMs, CRFs), Structured Perceptron, and RNNs will depend on the type and length of the sequences being processed, as well as the amount of data available for training the system. CRFs, for example, have a much lower computational cost than LSTMs and require less data for providing accurate predictions. RNNs and CRFs can also be combined by stacking a CRF over RNN outputs, thus providing advantages from both methods.

3.7 Automatic annotation of claim elements: Challenges

As with the document segmentation task, the claim segmentation was formulated as a list membership problem, in which each claim element is a list of words, and has a single type: its semantic function. Words are classified according to their element type and boundary condition: beginning, middle, end.

Different from document segmentation, however, the same type of element can appear multiple times in a claim, and a word can be part of multiple nested elements, each one with their own boundaries.

A proper representation scheme is needed for correctly classifying the words, thus identifying the elements and segmenting the claims. In this case, lexical, syntactic and semantic characteristics of the words in the sentence should be considered, as they are all influential in the reading and interpretation of a claim. Additionally, the classification method should also be able to deal with long sentences, as it is the case for claims.

3.7.1 The vocabulary issue

Due to the technical nature of patent text, there are cases when some vocabulary used is too uncommon and cannot be found in other texts or dictionaries. This is either due to such words being specialized and only known in a certain domain of knowledge, or because they are novel and were invented to convey the stated idea. In both cases, such words are typically formed by a process known in morphology (linguistics) as *compounding*, where a new word is obtained by combining two or more existing ones.

Those words are defined as out-of-vocabulary (OOV) and they cannot be represented by distributional-based methods, as they have no frequency information about them to work with. An alternative way of representation is needed to deal with them.

Examples:

"...the central axis and defining an *extrafilamentary* space within the housing..."
(patent EP-0915965-B1)

"...said secondary flows are *co-rotating* with the ducted fluid flow."
(patent WO-2001006134-A1)

In these examples, the words "extrafilamentary" and "co-rotating" are uncommon and considered OOV in this research corpora. OOV words are exceptional, but critical for correctly representing the idea of a claim element.

3.7.2 The semantic composition issue

Another important issue besides the word representation itself is the fact that claim elements are composed of multiple words, e.g., "reinforcing structural material" (Fig. 3.3). The composition of word meanings in such cases is important for the characterization of the element, and helps to disambiguate similar phrased ones.

Word embeddings are not appropriate for such composition issue, as dimensions in an embedding have no intrinsic meaning. Although TOP (Section 2.7) helps discriminating sentence roles, it relies on syntax constraints that are not applicable for short phrases.

3.8 Dealing with out-of-vocabulary words: Morphological decomposition

Managing the effective vocabulary is an important issue when dealing with large corpora. This can be done in several ways, the most common being excluding words below a certain frequency threshold. Excluding "*stopwords*": function words with little or no semantic value, such as determiners, is also done in Information Retrieval systems. Those approaches work well for general purpose text, but are inadequate for patent text, especially claims, due to their technical (and sometimes novel) vocabulary and constrained syntax. The syntax constraints make the phrase structure to be closely related to the semantic functions of the claim elements. Therefore, function words such as determiners, prepositions and conjunctions are very important and cannot be ignored.

Another issue relates to not only the size of the corpora, but to its rate of change, i.e., modifications or growth. The more dynamic a text corpus is, the more likely vocabulary

will change over time, requiring language models that are not static. In case this is neglected, most NLP tasks will get progressively less accurate and the systems will require constant re-training to keep useful.

One way of better managing vocabulary in a language model is to work from the most basic lexical units, in other words, from the morphological level. Such an approach has two major benefits: *(1)* it allows more efficient representation of common vocabulary due to the fact that basic symbols are far less numerous than their compositions, and *(2)* it enables representation of unseen terms, by attempting to decompose them in known symbols. While (1) is greatly helpful in achieving efficiency in large corpora, (2) is essential to deal with unpredictable changes in vocabulary. Patent corpora are both large and lexically unpredictable, thus making a morphological level approach advantageous.

3.8.1 Morphological decomposition

Morphological decomposition, also known as *morphological parsing*, is a linguistic process that deals with breaking-up words into their minimal meaning units, called *morphemes*. The morphemes are the basic written communication units in a language, so any given word can be expressed as a composition of morphemes. Such composition may happen in several ways (e.g., derivation, inflection, compounding) that are studied in the linguistics field of *morphology*. Some examples of morphological decomposition:

- "calling": composed by inflection
  - "call": stem (verb)
  - "-ing": inflectional suffix (tense)
- "unreliable": composed by derivation
  - "un-": derivational prefix (negation)
  - "rely": stem (verb)
  - "-able": derivational suffix (mode)
- "mice": composed by inflection
  - "mouse": stem (noun)
  - "+PLURAL": irregular inflectional suffix (number)
- "homestay": compound
  - "home": stem (noun)
  - "stay": stem (verb/noun)

The ability to perform such decomposition process is desirable because it allows interpretation of words without previous knowledge of them, if they are composed of identifiable morphemes. This can be easily observed by a human reader, upon finding some new word that is not in the dictionary, but whose meaning is obvious from its parts. For example, the word "extrafilamentary" is not listed in English dictionaries, and arguably does not need to be, as a fluent reader in English will understand its meaning by breaking

it into its three composing morphemes: [extra- + filament + -ary]. However, as can be seen in the example list previously shown, not all words are simple segmentation cases (e.g., unreliable has a character change). Additionally, some words have non-obvious decompositions due to irregular inflections (e.g., mice, feet), and some words may have multiple possibilities of decomposition (e.g., infamous = [in- + fame + -ous] or [infamy + -ous]) due to having multiple etymologies.

A morpheme level language model, thus need to be able to represent the morphemes and appropriate transformations that encode words. Additionally, it may be able to represent the composition processes taking place (e.g., inflection), in order to facilitate other NLP tasks that benefit from such knowledge, such as POS-tagging and dialogue generation.

3.8.2 Related work

Morphological decomposition processes have been object of study by linguists for a long time. In particular, a study on the quantification of morphological aspects of different languages can be seen in the work of Greenberg [33], where the concept of "index of synthesis" is described. A more recent study by Libben et. al. [34], focus on the psycholinguistic conditions for the understanding of words from their constituent morphemes, specially through the notion of "semantic transparency". The index of synthesis and semantic transparency are valuable tools for designing language models at morpheme level, since they can be statistically modeled and evaluated [34, 35].

Unfortunately, such attention to morphology is not given by the NLP research community, due to its inherent complications and lack of benefits in higher level tasks (e.g., parsing, sentiment analysis), where morphological information has quite less impact. NLP investigations using morphological information can be found in the works of Baud et. al. [36] in the medical domain, exploring morphological regularities found in electronic patient records; Thurmair [37] took special notice of the importance of case information when doing Statistical Machine Translation from languages with large differences in morphology. In more recent works, Konkol and Konopik [38] highlight the role of lemmas in Named Entity Recognition for the Czech language, due to its highly inflectional nature; Carneiro et. al. [39] highlights the strong correlation between the index of synthesis and the best architecture for POS-tagging a given language.

Recent efforts for morphological decomposition can be found in the works of Sakakini et. al. [40] in an semantically-driven, automatic rule inference method for extracting morphemes in English, Turkish and Finnish; and Faruqui et. al. [41] in a inflection generator based on a Sequence to Sequence (Seq2Seq) Recurrent Neural Network model with ensembling for German, Spanish, Dutch, French and Finnish languages. Sakakini et. al. [40] presented state-of-the-art results in the MorphoChallenge dataset [42], which is a current test collection for morphological decomposition.

The method developed in this work follows a similar principle of Faruqui et. al. [41], but not limited to inflections and instead of adopting a semi-supervised approach, uses a fully supervised one, made possible through the extraction of etymology information from Wiktionary [43].

3.8.3 Sequence to sequence morphological decomposition from etymology data

Obtaining morphemes: Extraction of etymology information from Wiktionary

One of the major problems faced when developing an automated method for morphological decomposition is the lack of examples covering all morphological composition processes. The reason is that collecting such linguistic data in a reliable way would take a large amount of professional linguists time, which means a high cost of production. Therefore, most methods are either rule-based or use a combination of unsupervised corpora with the scarce public material available for training and evaluating morphological data. One example of such professionally curated material is the MorphoChallenge dataset [42] which contains detailed information for less than 2000 words.

However, there is currently a much more abundant source of morphological decomposition data: the etymology section of Wiktionary entries. Since it provides etymology information, it also presents the most accepted decompositions for the entries were this information is available (Fig. 3.9). There are, however, some drawbacks:

- The information is community driven and curated. This means it is less reliable than professionally made datasets.
- There is no detailed morphological attributes, such as case and tense information.
- The information is semi-structured. So a specific parsing process is necessary to extract it.

Etymology [edit]

equi- + *-valent*. From Latin *aequivalentem*, accusative singular of *aequivalēns*, present active participle of *aequivaleō* ("I am equivalent, have equal power").

Figure 3.9: Example of etymology section for the dictionary entry "android". The contents are stored as a mix of free text and wiki markup.

Nevertheless, the amount of decomposition entries available is more than 100 times larger than the MorphoChallenge dataset, so an adequate approximation function might offset the lack of reliability in a minority of the samples, assuming a mostly correct dataset. This turns out to be a "textbook case" of Deep Learning application, which motivated the extraction and structuring of this information as a separate morphological decomposition dataset.

The process starts with obtaining the Etymology section from each Wiktionary entry and extracting only the morphological decompositions. This was done with the same parser used for extracting term definition for concept graphs in TDV (see Section 3.9), with few adjustments.

Now with a database of (word → decomposition) pairs, a manual inspection was done to identify and correct some common mistakes found in the annotations. Frequent mistakes benefited the most from automation applied via regular expressions.

Next, inflection information was added to the decompositions in the entries that had them. Ex: cats is correctly identified as an inflection (plural) of cat. Such information is not part of the etymology section. A simple rule-based method was developed for comparing the listed plural inflection and the word, then selecting the appropriate decomposition.

Finally, the morphemes in the database were normalized to the smallest ones available. The reason for this is that most decompositions in Wiktionary are not complete. For example: the word "unlockable" is decomposed as [un- + lockable], whereas a full decomposition would be [un- + lock + -able]. This was solved by expanding each morpheme through a multi-pass dictionary approach, were each morpheme was checked for possible further decompositions, until no further expansions were possible.

Seq2Seq Deep Learning architecture

The first decision regarding the processing of words as a sequence of morphemes is defining the basic element of the sequence. In this case, the trivial solution is to divide the words by their characters. Encoding each character as a vector can be done by one-hot coding, in which each symbol is given a different unit vector according to their index in the symbol list (Fig. 3.10).

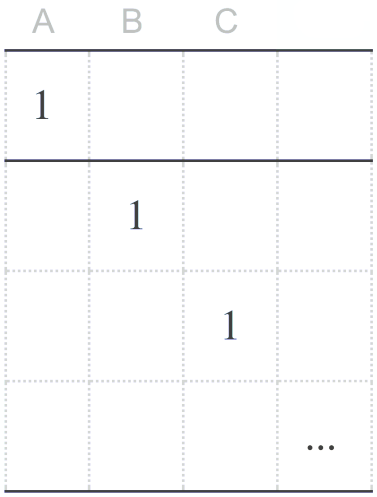


Figure 3.10: Illustration of the one-hot coding scheme. To each symbol (A, B, C, ...) is assigned a unit vector, where the non-zero element corresponds to the symbol's index.

The structural mapping of a word to its morpheme constituents is not as simple as a piece-by-piece separation, as observed in Section 3.8.1. In many cases the total number of characters in the decomposition will be different from the word, vice-versa. Therefore, the appropriate ANN architecture for modeling this problem is the sequence to sequence (Seq2Seq) RNN model. In this architecture, the sequence inputs are processed at each time step by a primary RNN, called "the *encoder*", which changes its internal state (hidden layer weights). The final state of this RNN encodes the entire sequence (hence the name) and is taken as input by a secondary RNN, called "the *decoder*", which proceeds to output on an arbitrary number of time steps, thus generating the output sequence. This scheme allows processing sequences of different length and is famous for its use in Neural Machine Translation, where the input is a sentence in the source language and the output is the translated sentence in the target language.

The Seq2Seq architecture for this problem can be seen as "translating" a word into its decomposed version. In this case, the output string of characters was separated by spaces, indicating morpheme divisions and dash markers ("-") were added to identify prefixes, suffixes, confixes and affixes from stems. The final architecture was composed of bi-directional LSTMs (Bi-LSTM) as both encoder and decoders, a Multilayer Perceptron as post-decoder layers, outputting in a softmax function layer: all output nodes sum 1.0, each one corresponding to the probability certain class being the correct answer. The network architecture is illustrated in Fig. 3.11.

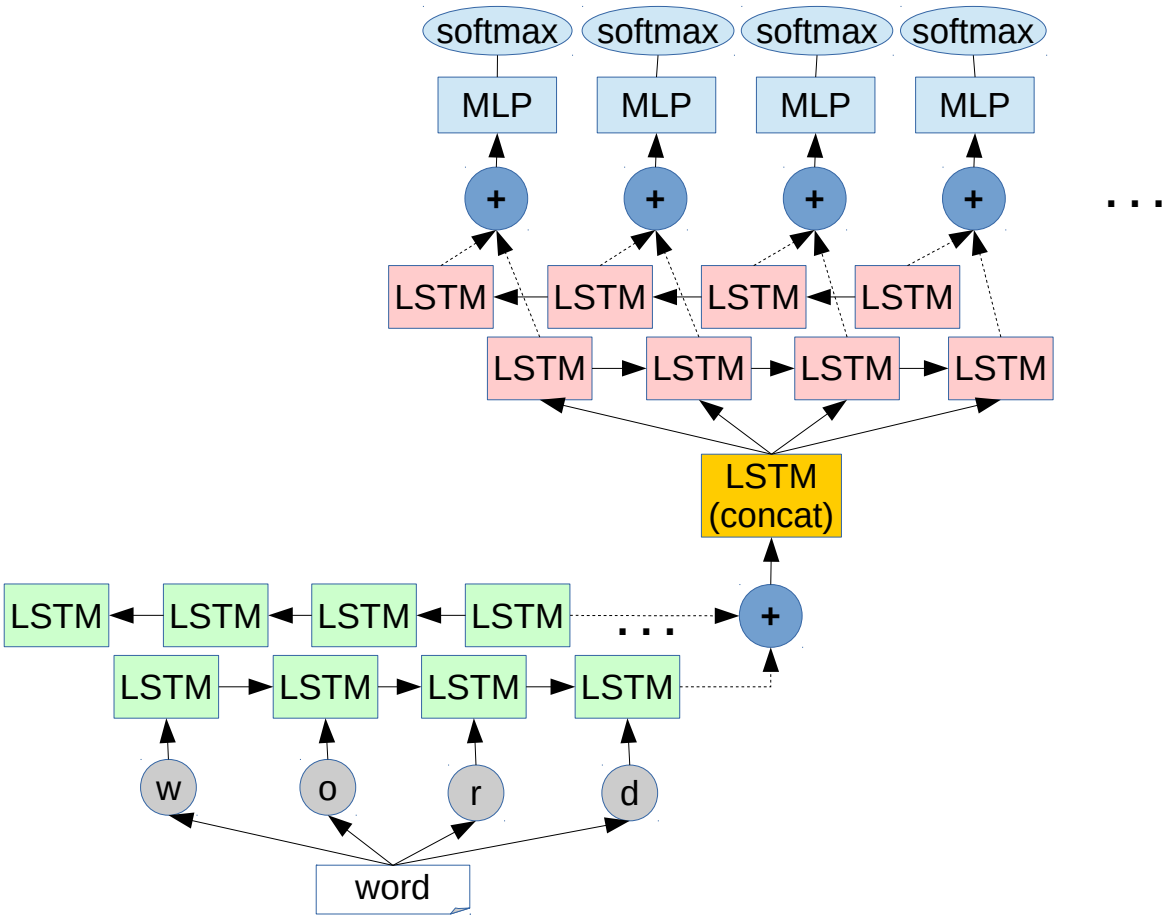


Figure 3.11: Sequence to sequence RNN architecture for morphological decomposition. The characters of the word to be decomposed are encoded as one-hot vectors and fed to the first Bi-LSTM as an input sequence. After all charachters have been processed, the final state of the encoder (the network's internal representation) is used as input for the decoder (the upper Bi-LSTM) followed by a Multi Layer Perceptron (MLP) and finally a softmax layer.

Confidence pruning for softmax

Using a softmax layer for the network output is a common solution for multi-class classification problems, although limited to cases where the number of classes is not large (up to the order of 10^2 as a general rule). However, a factor not so commonly exploited is that the probability distribution modeled by the softmax function can also be interpreted as a confidence measure both for the element it applies to and sequence-wide. In this interpretation the average of the maximum confidence values for each element is taken as a confidence measure for the sentence.

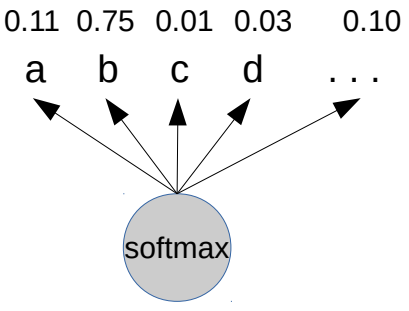


Figure 3.12: Softmax outputs.

With this confidence value, a threshold (or set of) can be defined to perform alternative operations on the sequence, should it cross the defined values. In the case of

morphological decomposition, a NOOP, i.e., the system
outputs the input sequence unchanged, is sometimes preferred over a low confidence decomposition. So a pruning threshold was defined so that if the confidence is below it, the network result is ignored and the input is returned unchanged.

Use of homogeneous ensembles with RNG selection

Recent advances in computer hardware provided software programmers with a variety of resources to distribute computing loads across multiple processor units. Those resources shall be exploited in Machine Learning systems not only for making them faster, but to improve their learning capabilities.

When considering the use of parallel computing units to solve a problem, the first concern is usually on effectively dividing the processing and data load. This is done in order to extract the maximum work from each unit, thus obtaining the best return of investment on equipment and programming hours. While this is also true for Machine Learning systems, there is another factor in ML that can take advantage of parallel computation: *non-deterministic optimality*.

Be it by design or by implementation choices, many ML methods have a source of non-determinism that affect the optimality of their results. In other words, running those methods several times on the same data and under the same external parameters (except for the source of non-determinism) will yield different results, some better than others. An ANN for example, has its weights initialized to random values before starting its training process.

Using ensembles is a way of mitigating such effect by having multiple simultaneous candidate answers, given the same input, and then trying to select the best one. Ensembles can be homogeneous (same method for all candidate answers) or heterogeneous (any combination of different methods). But while using ensembles have shown very good results recently, there was no explicit guarantee that a group would provide a better answer than a single learner, specially in homogeneous cases. However, a study on homogeneous group phenomena in computer game playing agents [44] indicates the possibility of steering the ensemble performance through informed selection of the random number generator distribution affecting the ensemble agents. The selection can be informed by testing the ensemble setup in a sample of the training set and choosing the distribution with best results, prior to training. The methodology and implications found in this study can be conveniently applied to the Seq2Seq morphological decomposition method by making the following analogies:

The player agent: An independent instance of the Seq2Seq network is analogous to a game playing agent and is a single member of an homogeneous ensemble of morphological decomposition agents.

The source of non determinism: The Seq2Seq network weights are initialized at random before training by a selectable initialization function. Such function was changed to generate values with uniform and gaussian distributions.

The decision process: A majority voting ensemble is run for each set of homogeneous Seq2Seq networks, in two modes:

- Overall: Votes are given for the entire result sequence.
- Character: Votes are given for each character of result sequence.

The first mode treats each resulting decomposition as analogous to a single move of the game playing group, while the second mode treats each element in the sequence as a separate move.

3.8.4 Experiments on morphological decomposition data

Data acquisition and preparation

The data for the experiments was obtained from the two resources discussed in Section 3.8.3:

- Wiktionary: etymology sections, inflection info and morpheme expansion.
  - Only morphemes.
  - 220K word decompositions at varying quality (overall good).
- MorphoChallenge 2010 database [42] (English section only):
  - Detailed morphological info: case, number, tense, among others.
  - 1K (training) + 694 (test) professionally decomposed words.

Experimental setup

For the performance evaluation, The Seq2Seq model was compared with three other systems which published results on the MorphoChallenge dataset:

- *Morfessor* (Kohonen et al. [45]): the original baseline for the MorphoChallenge 2010, using a probabilistic graphical model approach.
- *MORSE* (Sakakini, et. al. [40]): the current peer reviewed state-of-the-art in the English MorphoChallenge. Baseline for this study.
- *Morfessor 2.0* (Virpioja et al. [46]): improved version of Morfessor. Featured in a technical report, indicating a very high performance.

The evaluation metrics used were the same as the MorphoChallenge:

Precision:
"`A number of word forms will be randomly sampled from the result file provided by the participants; for each morpheme in these words, another word containing the same morpheme will be chosen from the result file by random (if such a word exists). We thus obtain a number of word pairs such that in each pair at least one morpheme is shared between the words in the pair. These pairs will be compared to the gold standard; a point is given for each word pair that really has a morpheme in common according to the gold standard. The maximum number of points for one sampled word is normalized to one. The total number of points is then divided by the total number of sampled words.`"

Recall: "`Recall is calculated analogously to precision: A number of word forms are randomly sampled from the gold standard file; for each morpheme in`

these words, another word containing the same morpheme will be chosen from the gold standard by random (if such a word exists). The word pairs are then compared to the analyses provided by the participants; a point is given for each sampled word pair that has a morpheme in common also in the analyses proposed by the participants' algorithm. Points per word is normalized to one and the total number of points is divided by the total number of words."

The F-score follows formula:

$$F{-}score = \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{3.1}$$

Also:

"For words that have several alternative analyses, as well as for word pairs that have more than one morpheme in common, normalization of the points is carried out. In short, an equal weight is given for each alternative analysis, as well as each word pair in an analysis. E.g., if a word has three alternative analyses, the first analysis has four morphemes, and the first word pair in that analysis has two morphemes in common, each of the two common morphemes will amount to 1/3*1/4*1/2=1/24 of the one point available for that word."

Two additional tests were performed, to check the impact of the large training data in the results, contrasted with the impact of using the proposed Seq2Seq architecture:

Levenshtein mappings test:

For each word in the test set, the Levenshtein string distance for each word in the training set is calculated. Next, the closest match is selected and its decomposition mapped to the test word as output.

This is a fast, lexical only method for verifying if there are enough similar matches in the training set for it to become a memorization task.

Word2vec + Levenshtein mappings test:

For each word in the test set, calculates the word2vec similarity for each word in the training set, falling back to Levenshtein distance if the word is not in word2vec trained vocabulary. Next, the closest match is selected and its decomposition mapped to the test word as output.

This test uses distributional semantic info from w2v embeddings primarily, making the (inverted) assumption that close meaning words may have similar morphology.

The Seq2Seq decomposition system was configured as follows:

- Layers and dimensions: input size = 40x32, BI-LSTM(encoder = 400, decoder = 300), MLP(1000), output size = 68x32.
- Using input masking, output sample weighting (per character).
- Dropout = 0.25 (encoder), 0.25 (decoder).
- Epochs = 25, batch size = 200.

- Confidence threshold = 0.8.
- Pruning: ON and OFF.
- Ensemble size ≤ 20.
- Word2Vec $d = 300$. (GoogleNews vectors)

Results & Discussion

The experiment results are presented in Table 3.1

Table 3.1: Performance results for the morphological segmentation test from MorphoChallenge 2010. Modification symbols to the method here presented were added for conciseness: [S: SINGLE, E: ENSEMBLE, U: UNIFORM, N: NORMAL, P: PRUNING]

Method	Precision	Recall	F-score
Morfessor (Kohonen et al. [45])	0.6562	0.6928	0.6740
MORSE (Sakakini, et. al. [40])	0.8198	0.6157	0.7032
Morfessor 2.0 (Virpioja et al. [46])	0.8995	0.6638	0.7639
This method (S, U)	0.7855	0.7072	0.7443
This method (S, U, P)	0.8156	0.7067	0.7573
This method (S, N)	0.8276	0.7383	0.7804
This method (S, N, P)	0.8512	0.7578	0.8018
This method (E, U)	0.8119	0.7204	0.7634
This method (E, U, P)	0.8359	0.7379	0.7839
This method (E, N)	0.8354	0.7451	0.7877
This method (E, N, P)	**0.8705**	**0.7615**	**0.8124**
Levenshtein mapping	0.5667	0.6412	0.6017
word2vec + Levenshtein	0.5627	0.6444	0.6008

The results indicate that collecting a large amount of morphological data allowed effective training of a Seq2Seq Deep Learning model for morphological decomposition. Additionally, using softmax maximum probability values as a confidence measure for pruning the outputs showed verifiable beneficial results, improving on state-of-the-art and approaching the technical report results.

The homogeneous ensemble approach significantly improved performance, and exploiting the RNG selection takes a large margin over the state-of-the art and even over the improved technical report results. Additionally, performance relative to the ensemble size was also measured, as shown in Fig. 3.13. Performance had an upward trend with the size of the ensemble, until stabilizing around the expected value for that particular configuration.

The Levenshtein and word2vec experiment results also indicate that while the large amount of training data was important, it was not a determining factor in the results. Therefore this experiment provided a good test case for the use of Deep Learning, in providing a powerful way of modeling decomposition functions. Nevertheless, the Morfessor [46] system also provides a valuable example in expert modeling of rule-based systems, which can perform well despite the limitations in available data.

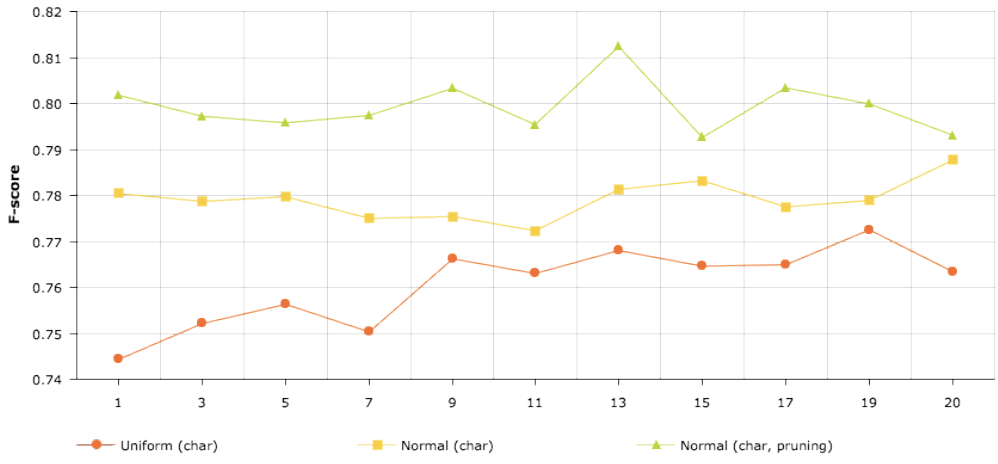


Figure 3.13: Performance x ensemble size for different random initializations of the Seq2Seq RNN homogeneous ensemble.

3.9 Dealing with semantic composition: Term Definition Vectors

A critical component in the goal of comparing claims elements is the ability of distinguishing terms (words, phrases), as well as the entire element, based on their meanings. For example, the word "locking" in the elements "an infrared signal for locking and unlocking the door" and "operation for locking a carrier frequency" has different meanings and should be associated and accordingly for meaningful comparison. For this purpose, a semantic representation should have the following set of capabilities:

- Allow meaning composition of multiple terms.
- Be able to carry information about multiple meanings for a single term, so they can be disambiguated.

3.9.1 Related work

In order to address the limitations of the most popular representation schemes, approaches for all-in-one representation models were also developed [47, 48]. They comprise a combination of techniques applied over different data sources for different tasks. Pilehvar [47] presented a method for combining Wiktionary [43] and Wordnet [49] sense information into a semantic network and a corresponding relatedness similarity measurement. The method is called ADW (Align, Disambiguate, Walk), and works by first using a Personalized PageRank (PPR) [50] algorithm for performing a random walk on the semantic network and compute a *semantic signature* of a linguistic item (sense, word or text): a probability distribution over all entities in the network where the weights are estimated on the basis of the network's structural properties. Two linguistic items are then aligned and disambiguated by finding their two closest senses, comparing their semantic signatures under a set of vector and rank-based similarity measures (Jensen–Shannon divergence, cosine, Rank-Biased Overlap, and Weighted Overlap). ADW achieved state-of-the-art performance in several semantic relatedness test sets, covering words, senses and entire texts.

3.9.2 Definitional semantics: Term Definition Vectors

In an attempt to offset the perceived weaknesses of the distributional methods, an alternative way of approaching the representation issue was developed. Taking a both linguistic and epistemic view, the basic premise of this language representation model is to represent knowledge as a set of individual concepts that relate to one another and are related to a set of terms. This idea is closely related to the Ogden/Richards triangle of reference [51] (Fig. 3.14), which describes a relationship between linguistic symbols and the objects they represent. The following notions are then defined:

- *Concept*: The unit of knowledge. Represents an individual meaning, e.g., cold (as in "low temperature condition"), and can be encoded into a term (symbol). It corresponds to the "thought or reference" from the triangle of reference.
- *Term*: A unit of perception. In text, it can be mapped to fragments ranging from morphemes to phrases. Each one can be decoded into one or more *concepts*. Stands for the "symbol" in the triangle of reference.
- *Definition*: A minimal, but complete explicitation of a concept. It comprises the textual explanation of the concept (sense) and its links to other concepts in a knowledge base, corresponding to the "symbolizes" relationship in the triangle of reference. The simplest case is a dictionary definition, consisting solely of a short explanation (typically a single sentence), with optional term highlights, linking to other dictionary entries.

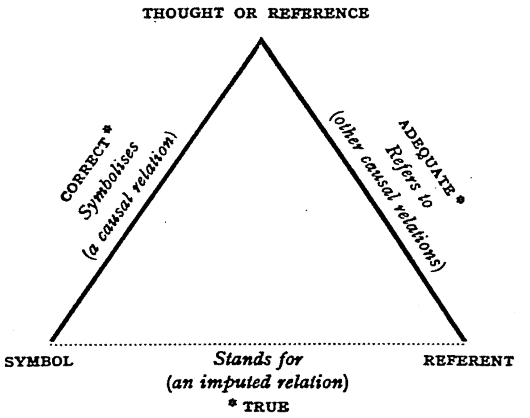


Figure 3.14: Ogden/Richards triangle of reference, also known as *semiotic triangle*. Describes a relationship between linguistic symbols and the objects they represent. [51]

The proposition is to define the meanings first and then associate the corresponding terms. In this notion, meanings are explicit and need only to be resolved, i.e., disambiguated, for any given term. Concepts are thus represented by prior definitions instead of distributions over corpora, hence the name"definitional semantics". Realization of such proposition was achieved through the following steps:

1. Formalization of concepts: The basic unit of knowledge is formalized as a function $\boldsymbol{c : S(\mathcal{L}) \rightarrow DG_w}$, where $S(\mathcal{L})$ is the set of senses in a linguistic resource $\mathcal{L}$ and DG_w is the universe of weighted directed graphs. $\boldsymbol{c(s)}, s \in S(\mathcal{L})$ is a lexical/semantic graph called *concept graph*, where a main addressing term, the root node, is connected to other terms through a set of edges. Each edge denotes a different type of lexical/semantic relationship, e.g. prefixation, synonymy/antonymy, and is directed from the root node to

the addressed terms. The edges are also weighted, denoting the intensity of a relationship. Fig. 3.15 illustrates a concept graph.

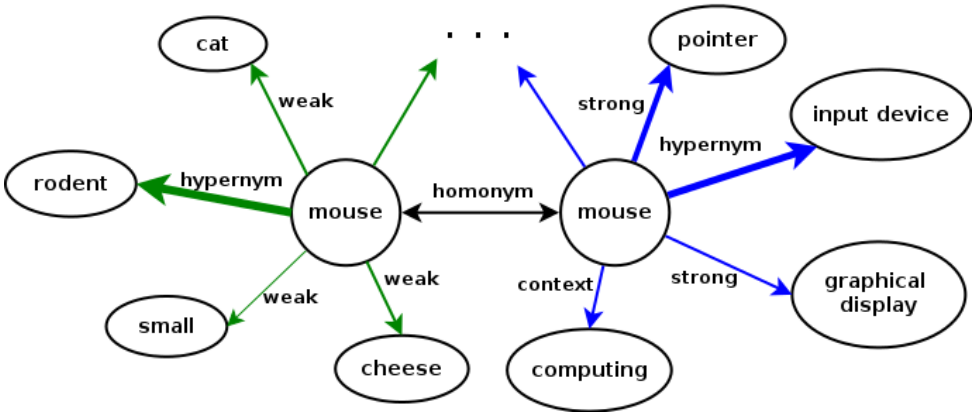


Figure 3.15: A simplified visualization of two concept graphs for the term "mouse". The leftmost one denotes the concept of the small rodent and the other denotes the computer input device. The edge labels represent the relationship type and the thickness represent the its intensity. [52]

2. Information extraction from the linguistic resource: *Wiktionary* [43] was used as the single linguistic resource $\mathcal{L}$. Wiktionary is a collaborative lexical resource, comprising millions of vocabulary entries from several languages. It includes contextual information, etymology, semantic relations, translations, inflections, among other types of information for each entry. Its contents are actively curated by a large, global community.

The data available from Wiktionary is semi-structured, composed of a set of markup documents, one for each vocabulary entry, following a standard of annotations for each language covered. All the documents are included in a single "database dump file" (XML format). In order to extract the linguistic information, an application was developed to convert the markup into a structured (JSON + schema) representation [53]. The structured data was optimized for the retrieval of Wiktionary senses and link types were categorized to produce concept definitions.

The information extraction procedure is divided in two stages: parsing and information extraction. Parsing takes as input an English Wiktionary database dump file and is done in three steps:

1. Count the number of entries and register their file offset.
2. Create a specified number of parsing processes and assign a group of entries for each process.
3. Collect and merge the results of each parsing process to generate the output JSON file.

Each parsing process takes as input Wiktionary entries, which are single XML elements, containing metadata and the contents of the entry (part-of-speech and sense data,

examples, etymology, among others) in *Wiki markup* [3] format.

The entry contents are parsed line-by-line, in event-driven fashion. Each line is checked for identifying patterns through regular expression matching, and upon finding a relevant pattern, the corresponding routine for extracting formatted data is called. For each entry, a JSON document is created with the structure illustrated in Figure 3.16.

After all entries have been processed, the parser results are merged into a single document list and sorted by the entry title. This document list is the parsing final output.

The information extraction stage takes as input the JSON output from the parser and is done in a single step. It uses the description, link markup, morphological and semantic information from each sense in each entry to produce the concept graph. A concept graph is represented by a $M^{L \times T}$ matrix called *concept definition*, where L is the number of link types and T is the vocabulary size. The link intensities are defined for each type, by multiplying a manually defined constant *link_base* (a model parameter) by the TF-IDF score calculated for the vocabulary with respect to the type. Figure 3.17 illustrates the process. The matrix is then flattened by concatenation of its rows (vocabulary dimension) to obtain *concept vectors*. Table 3.2 describes the link types used. Figure 3.18 illustrates the entire information extraction process flow.

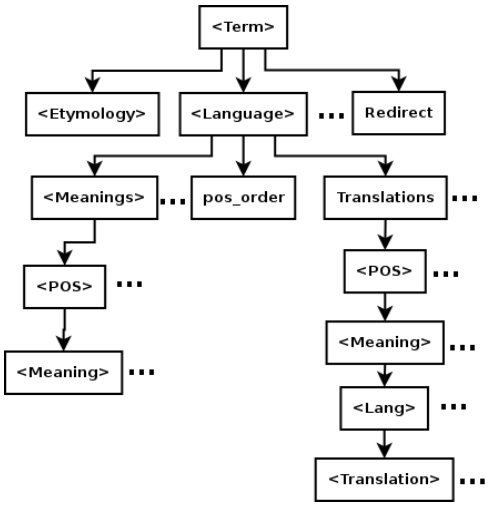


Figure 3.16: Wiktionary entry hierarchical structure obtained from the parser. [53]

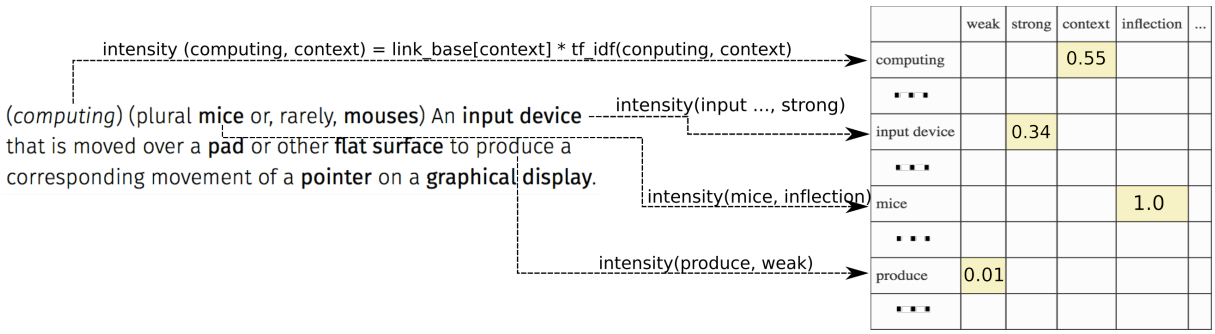


Figure 3.17: Representation of one Wiktionary sense definition for "mouse" as an encoded matrix: the *concept definition*. Each Wiktionary link is categorized and mapped to a vector space. [52]

3. Definition of a term as a composition (mixture) of concepts: For a given term, its concept vectors can be combined (summed) to represent ambiguity (Fig. 3.19). A combination of concept vectors is called *term definition vector* (TDV), which gives the name of this approach. The concept vectors are cached in memory or disk and are indexed by their entry title and POS or by a hash based sense index, so they can be quickly retrieved.

[3] `https://en.wiktionary.org/wiki/Help:Wikitext_quick_reference`

Table 3.2: Link types used for the construction of concept graphs. They comprise both lexical (morphology, etymology) and semantic relationships between the root term, i.e., the Wiktionary entry title, and the terms used to describe the meaning.

Type	Description
weak	A term included in the description of the meaning on the Wiktionary entry.
strong	A term linked to another entry, i.e. a {highlight}, included in the description of the meaning.
context	A Wiktionary context link, explaining a specific situation in which the meaning described occurs.
synonym	A synonym relation. If it is an antonym, the sign of the link is reversed.
hypernym	A hypernym relation.
homonym	A homonym relation.
abbreviation	If the meaning described is given by interpreting the root term as an abbreviation.
etymology	Used to describe the origin of the root term of this meaning.
prefix	Denotes a prefixation (morphological) relationship of the root term.
suffix	Denotes a suffixation relationship. Same as above.
confix	Denotes a confixing relationship. Same as above.
affix	Denotes an affixation relationship. Same as above.
stem	Denotes a morphological stem relationship of the root term.
inflection	Denotes an inflectional relationship of the root term.
translation	Denotes a translation link relationship (target language) to the root term (source language).

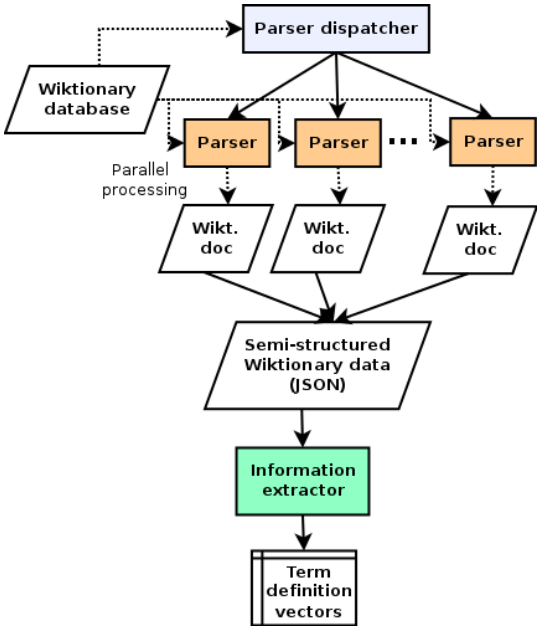


Figure 3.18: Flow of the information extraction process. [53]

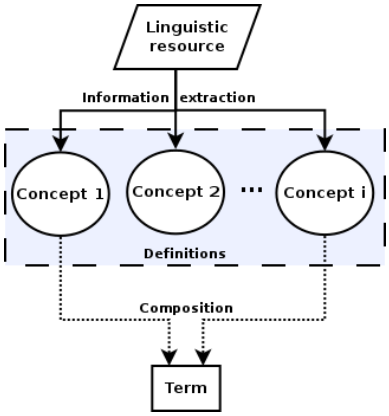


Figure 3.19: Process of definitional representation. Given a set of concepts obtained from a linguistic resource, a term can be defined as a composition of concepts. A term is said ambiguous if it is composed by more than one concept. [52]

Wiktionary entries also cover foreign terms, listing senses in the source language, e.g., English meanings of the French word "rouge" in the English language section. Definitions for these terms are also included into the concept definition set. Additionally, translation links are provided for many sense definitions. Such links, as well as term redirections, i.e., distinct terms pointing to the same Wiktionary entry, are mapped to a single concept. This allows foreign terms to take advantage of the same concept graphs as the source language equivalents.

Similar to the text embeddings obtained from distributional methods (i.e., distributed

representations), term definition vectors can be used to calculate semantic similarity and relatedness through *cosine similarity*. Therefore, experiments were conducted to compare effectiveness in such tasks (Section 3.9.3), revealing a complementary relationship between definitional and distributional approaches.

However, there are some important differences between the two approaches:

- Vector type: While text embeddings provide *dense* vectors (i.e., most values are non-zero), TDVs are sparse representations (i.e., most values are zero). Furthermore, the dimensions of distributed representations have no particular meaning, while TDV dimensions are interpretable: each one represents a lexical/semantic relation type from a sense to a term in the vocabulary. This means that the latter can be used as a human-readable feature set, from which specific feature groups may be selected depending on the desired use. This characteristic is exploited in solving morphological decomposition (Section 3.10.2) and segmentation of patent claims (Section 3.10.3).
- Cosine range: TDVs admit negative link weights, such as in the antonym relations. This means that the vectors are not limited to a single orthant in their vector spaces, as word2vec, and cosine can range from -1.0 to 1.0. Fig. 3.20 illustrates this idea.

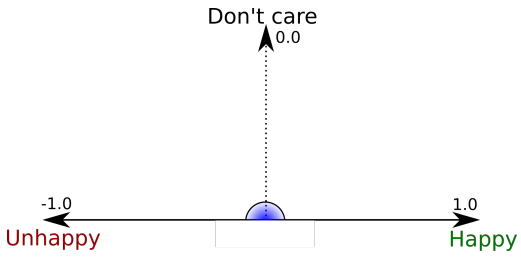


Figure 3.20: Bi-dimensional cosine measure representation for term definition vectors regarding the term "happy". Terms with negative cosine have opposite meaning, while those with close to zero are unrelated.

3.9.3 Experiments on semantic similarity and relatedness data

Data acquisition and preparation

The term definition vector representations obtained in this work were evaluated in the *SimLex-999* test collection for semantic similarity benchmark [54]. This test collection contains a set of 999 English word pairs, associated to a similarity score given by a group of human annotators. The set is divided sets of in pairs according to Part-of-Speech (POS): 666 noun pairs, 222 verb pairs and 111 adjective pairs. The POS information allows partial or complete disambiguation of the definition vectors. The choice of SimLex-999 was due to the type of similarity measured by this set, which excludes relatedness and is closer to the type of information captured by the concept definitions. Additionally, the *WordSim-353* [55], *RG-65* [56] and *MEN* [57] test collections for semantic relatedness were also included in the evaluation, to verify the representation performance in measuring relatedness. While the MEN test collection also includes POS information, WordSim-353 and RG-65 do not include it, so sense distinction was not applied for the latter two. Furthermore, the test collections used in this work do not contain foreign words, so the translation-link features mentioned in Table 3.2 are only presented as part of the method's description, but are not evaluated.

Experimental data and model parameters were set as follows:

- Linguistic information source: Wiktionary English database dump (XML + Wiki markup), 2015-12-26, containing more than 4 million entries. A reduced set, with only English, French, Greek, Japanese, Latin and Vietnamese language entries was used in the experiments. This set had about 734K entries, from which approx. 1 million concepts where extracted.

- *link_base* constants were set as: $weak = 0.2$, $strong = 2.0$, $context = 0.5$, $synonym = 10.0$, $hypernym = 5.0$, $homonym = 7.0$, $etymology = 1.0$ (also applied to morphological links) and $pos = 1.0$. The constants were adjusted by increasing or decreasing their values individually in intervals of 0.2, and observing the effect in ρ for SimLex-999 in the first fold of the cross-validation. The optimal values were selected and kept constant for the remaining folds and for the other tests. This was done because changing *link_base* for each fold would create an unrealistic use scenario for our system, which cannot change *link_base* online. The cross-validation was repeated two times, with very minor differences between both test runs. The constant values reported here are from the last run.

- SVM^{rank} was set with a default linear kernel and C parameter (training error trade-off) was set to 8 for MEN and 5 for the other test collections. The value was increased in unit intervals, until convergence was longer than a time threshold (10 minutes). This parameter was adjusted using the training set for MEN, or inside each CV fold for the rest.

- Both *Word2Vec* and *GloVe* were used with pre-trained, 300-dimensional models: 100 billion words GoogleNews corpus and Common Crawl 42 billion token corpus respectively.

Experimental setup and methodology

Evaluation is done by computing the Spearman's rank correlation coefficient (ρ) between the human annotators' similarity or relatedness scores and the scores given by the automated methods. A value 1 for ρ means a perfect match between the relative positions of the pairs, when ranked by their similarity scores.

For the SimLex-999 test, the cosine similarity between the term definition vectors was set as the similarity score. For the WordSim-353, RG-65 and MEN tests, the absolute value of the cosine similarity was used instead, since opposite words are related.

A 10-fold cross-validation test using random pairs without replacement was run for the entire sets (5-fold for RG-65), except MEN. The MEN test collection is separated into training and testing sets, with 2K and 1K word pairs respectively, so these were used in place of the cross-validation. For each fold, the ranking scores provided by the trained ranker were used as similarity scores for calculating ρ. The average of all folds was considered the final result.

dLCE [58] was chosen as baseline, for being the best single information source method in the SimLex-999 test collection. Further results include Recski et. al. [59] (state-of-the-art), Ling Dense [41], Word2Vec [9], and GloVe [10]. For WordSim-353, GloVe, Word2Vec, Ling Dense, and ADW [47] were included. For RG-65, Ling Dense, GloVe, and ADW (state-of-the-art), were included.

Combining Distributional and definitional approaches

The complementary nature of the distributional and definitional methods revealed in early experiments motivated further investigation about the possibility of combining both. An additional test was performed to explore the possibility of combining distributional and definitional approaches. In this test, a small set of features was created to train a Learning-to-Rank model, in order to improve the similarity scores. The features were as follows:

- Presence of synonym, hypernym, strong and weak links [4] between the pair of words. Each link type is a separate feature.
- Term definition vector cosine similarity.
- word2vec cosine similarity.

The features were computed for each pair and passed to SVM^{rank} [60] for training and validation.

Results & Discussion

Results of the experiments are presented in Table 3.3, where they are compared to the other methods.

Table 3.3: Performance of different methods for the SimLex-999, WordSim-353, RG-65, and MEN test sets, reported as Spearman's rank correlation coefficient *rho*. The methods marked with ⋄ use a single information source. Fields marked with "-" indicate that the results were not available for assessment.

Method	**ρ@SimLex-999**	**ρ@WordSim-353**	**ρ@RG-65**	**ρ@MEN-1K**
Word2Vec (W2V) ⋄	0.38	**0.78**	0.84	0.73
GloVe ⋄	0.40	0.76	0.83	-
Term Def. Vectors (TDV) ⋄	0.56	0.36	0.68	0.42
Ling Dense	0.58	0.45	0.67	-
dLCE ⋄	0.59	-	-	-
TDV + W2V + SVM^{rank}	**0.62**	0.75	0.72	0.78
Recski et. al. [59]	**0.76**	-	-	-
ADW	-	0.75	**0.92**	-

The results indicate that in the semantic similarity test, the term definition vectors perform closely to other representation models taking advantage of human curated data (e.g., WordNet). It also outperforms the most popular distributed representations. However, they are largely outmatched in the semantic relatedness test, in which the distributional approaches show superior performance.

An interesting observation can be made when combining word2vec similarity with term definition features through the use of Machine Learning. A performance trade-off seems to exist at the semantic relatedness tests, but such trade-off does not happen in the similarity test. This allowed the combined model to improve considerably at little cost. Further analysis helped in understanding the reason for this particularity (Section 3.9.3).

[4] See Table 3.2

The experiments have also shown that the method for extracting concept definitions is not computationally expensive. The developed implementation took about 6 minutes to extract all concept definitions from the structured Wiktionary data used in the tests, using a modern desktop computer (3GHz processor and at least 8GB RAM). Structuring Wiktionary data took less than 20 minutes with the same equipment, and was done a single time.

TDV code, data resources and demonstration can be obtained in the open repository `https://github.com/dscarvalho/tdv`.

Error analysis

A fundamental step in improving a new method and better understanding the problem it tries to solve is to search and identify its flaws. With this in mind, a detailed observation was done on the error cases identified in measuring similarity from the SimLex-999 set. In this analysis we considered as error any word pair that was put among the top 15% similarity scores by the human annotators, but was ranked in the lower 50% using the definition vectors. The same applies for the bottom 15% scored by humans, that are ranked in the upper half by the TDV approach.

The errors found were classified in four categories:

- Insufficient links in Wiktionary: this type of error occurs when the wiktionary sense corresponding to a concept lacks annotations. Typical cases contain only a short description, with no links. The concept graph is then left with only weak links, which have little impact on similarity calculation. The pair *drizzle–rain* (noun) is one example of this.
- Undeclared hypernymy: certain cases of hypernymy are not solved in the concept extraction, since they require multiple hops in the definition links to be found. The pairs *cop–sheriff* and *alcohol–gin* (noun) are instances of such problem.
- Casual vs. formal language semantics: not a flaw in the method per se, but an error caused by the differences in formal description of a language (in a dictionary), when compared to casual use. The pair *noticeable–obvious* (adjective) illustrates this.
- Other: flaws in the extraction process or annotation problems in Wiktionary.

Those errors affect the pairs in the top 15% human similarity scores 7 times more than the lower 15%. They are distributed as shown in Table 3.4.

Table 3.4: Distribution of definition vector error types in SimLex-999.

Type of error	**Proportion**
Insufficient links	21.4%
Undeclared hypernymy	38.1%
Casual semantics	14.3%
Other	26.2%

Having about one quarter of the errors in the "other" category shows that there is space for improvement in the concept extraction process. The insufficient links and undeclared hypernymy categories are cases in which distributional approaches may do better if similarity is high, due to these words intrinsic relatedness.

Analysis of SVM^{rank} scores showed that the insufficient links category benefited the most from the combination with word2vec. This is due to the features chosen for use with the ranker made such cases distinguishable and more likely to receive a larger weight from the word2vec similarity score after training. The undeclared hypernymy cases, on the other hand, are not so evident and would require a more complex approach on the concept extraction process.

3.10 Claim segmentation flow

The core aim of the semantic segmentation of patent claims is to obtain a dynamic mapping function for *claim element word* $\Rightarrow$ *semantic function + boundary. semantic representation* of an element can be done by combining word representations into one representing the element's meaning, after segmentation.

Different from document segmentation, the intermediate states of the sequence processing are very important, since they model the segment's meaning. In other words, the information on the internal state of the system as valuable as the identity of the sequence elements. This means the use of a Structured Perceptron or a pure probabilistic graphical model (HMM, CRF) is out of question, leaving an RNN as immediate alternative.

As described in Section 3.7, a word in a claim can be part of multiple nested elements, so this is a multi-class labeling task, where the classes have a determined hierarchy (e.g., $CLAIM\_REF\_NUM$ is a subclass of $CLAIM\_REF$). This can be approached as a joint label (i.e., composite classes) or joint model (i.e., single classes with simultaneous or sequential classification) sequence classification problem. For simplicity, the former was chosen as a way of obtaining a proof-of-concept model, which can then be improved by choosing the latter.

The flow of information in the claim segmentation method was designed as a multi-modular pipeline, where the output of one module serves as input for the next ones, and annotations are included incrementally in the sequence, from the lowest to the highest level of abstraction. It starts by reading the sentence words, performing morphological decomposition, then using the morpheme information to obtain Part-of-Speech tags, followed by including semantic representations (distributed, definitional) based on the morpheme and POS info. Finally, the morphological, POS, and semantic representations are passed as inputs for the semantic segmentation tagger, which can build its own internal representation based on the cumulative information obtained in the previous steps. Fig. 3.21 illustrates the flow of information in the system. Each step in the pipeline was implemented as a separate RNN, that was trained independently. While in theory the modules can be jointly trained, this is not done due to its practical difficulty (computer/time resources and input mappings from different corpora). Other linguistic attributes, such as syntactic dependencies, were also considered to be included, but discarded early due to the difficulty of reliable parsing on very long sentences.

Input and output information is managed using the *Simple Annotation Framework*, a tool created for facilitating data exchange between different NLP annotation systems.

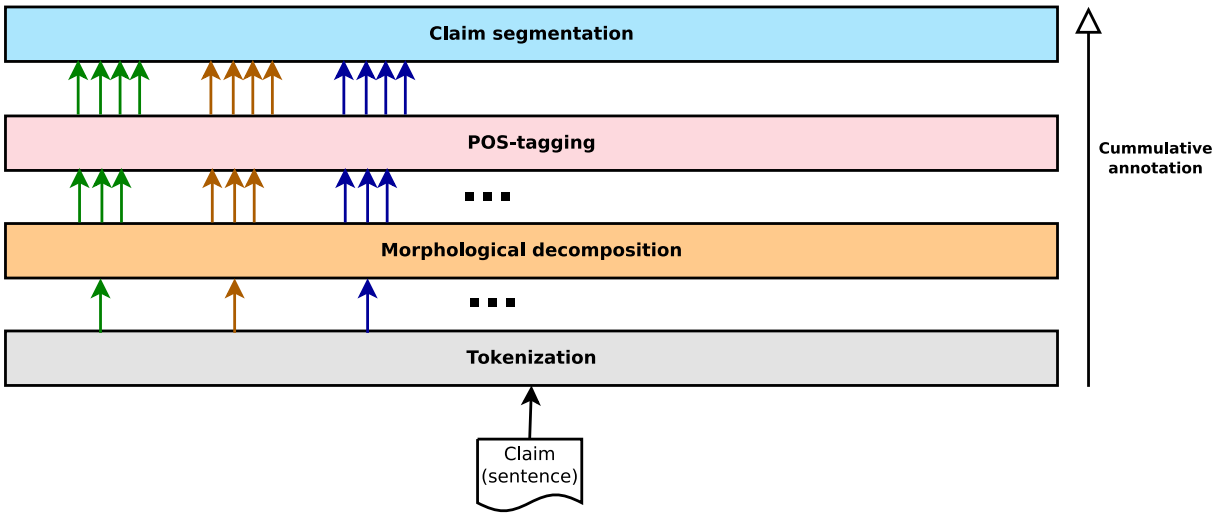


Figure 3.21: Anotation flow overview for the claim segmentation pipeline. Each module outputs its own information, in addtition to its inputs. Each module is a separate, independent RNN.

3.10.1 The Simple Annotation Framework

A secondary problem found during the course of this research was the integration issues between different NLP tools. They are developed using different programming languages, using different annotation standards and formats.

To facilitate the work with a variety of NLP tools, and also integration between the claim segmentation system RNN modules, the Simple Annotation Framework was developed. It consists on a library written in python language, that provides facilities for collecting data from any NLP annotation tool and outputting such data in a variety of formats (e.g., CoNLL, TSV, JSON). However, it provides its own internal representation of the data, which can be consumed by other applications through a common application programming interface (API). The internal data model is flexible enough to be used for most types of linguistic annotation, and can store other types of data associated to the language items (e.g., statistics, data sources, schemas, etc.) Fig. 3.22 shows a simplified diagram of the framework.

The data model consists in four main classes: *Document*, *Sentence*, *Term* and *Token*, derived from a single *Annotable* abstraction, that implements an annotation map "name"→"object" accepting any kind of annotation object, including character strings, lists, maps, among others. The model is hierarchical, where a document contains a list of sentences, and a sentence contains a list of terms (n-grams) and tokens (unigrams).

SAF is used to record the outputs of each module to each token and also to collect statistics from the modules internal measurements, allowing some level of explanation about the outputs.

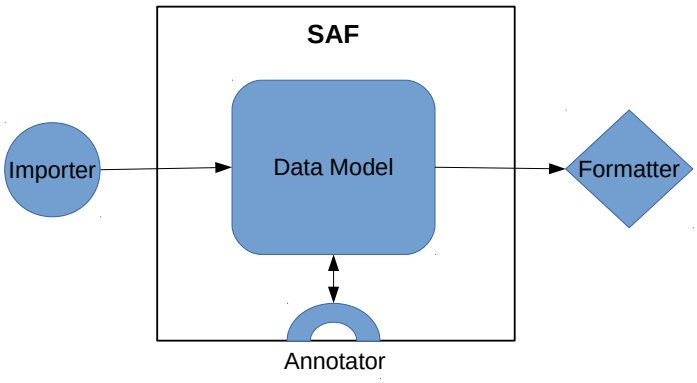


Figure 3.22: Simple Annotation Framework component diagram. Importer modules are used to collect data from other NLP tools, converting it to the internal data model. The data can then be exported using the formatter modules. An annotator interface is used to manipulate the internal data model, including, or modifying annotations.

3.10.2 Application of morphological decomposition

Morphological decomposition annotations are used on both the Part-of-Speech tagger and claim segmentation modules.

In the POS-tagger module, they substitute the surface form of the words by space-separated morpheme tokens, and for each morpheme (limited at four) a prior POS list is obtained from the TDV attributes and used as inputs for the RNN.

In the claim segmentation module, the morphemes are used for out-of-vocabulary (OOV) words, where the TDV representations for each morpheme are summed and used as the OOV word representation.

3.10.3 Application of Distributional & Definitional semantics

There are 3 attribute sets used in the claim segmentation module of the ANN architecture: POS, word2vec and TDV.

The POS attribute is a single one: each POS class is encoded as a one-hot vector and passed directly to the network.

The word2vec attribute set comprises all dimensions of a word2vec embedding from a file trained in the entire EPO corpus, minus the documents contained in the reference corpus.

Besides allowing combination of morphemes and word representations, TDV provides indexed links of lexical and semantic relationships (the TDV dimensions) that are not available in distributional representation methods, such as word2vec.

In claim segmentation, this translates into a set of attributes representing the "strongest" links of each word in a claim. Words with similar definitions are likely to have matching attributes. Those are an alternative and complementary set of attributes to the Neural Network.

The TDV attribute set is actually a combination of the TDV links and word2vec, along with TDV link types, which is done in the following way:

- The k highest valued links are selected from the TDV vector for the given word, disambiguated by POS. If there is no POS-tag, no disambiguation is done. If the

word is OOV, the TDV of each morpheme (if available) is summed before ranking the links.

- The links are then alphabetically sorted by their indexing term.
- The word2vec embedding is obtained for each indexing term and multiplied by the corresponding TDV link value. If no embedding is available, a zero vector is used instead.
- The embeddings are concatenated in a single vector.
- TDV link types (15 in total) for each corresponding position in the sorted list are encoded as one-hot vectors.

This setup allows the use of TDV attributes as fixed-length inputs for the RNN.

3.10.4 Neural Network architecture

The RNN modules are set up in the following order:

1. **Morphological decomposition**: Seq2Seq Bi-LSTM Neural Network. Takes character-based encodings of words as inputs and provides morpheme lists as outputs.
2. **POS-tagger**: Bi-LSTM-CRF Neural Network. Takes pairs (*word*, *attributes*) as inputs, where word is a surface form or space-separated morpheme token list, and attributes is a list of the following:
   - A prior list of POS classes for the input word.
   - A prior list of POS classes for each morpheme the input word (limited to 4), if available.

   word is character encoded and all POS classes are one-hot encoded and concatenated in a single vector. Outputs a single POS-tag (Universal POS [61]) for each word in the input.
3. **Claim segmentation**: Bi-LSTM-CRF Neural Network. Takes the inputs listed in Section 3.10.3.

   Two output configurations were proposed, both represented as a one-hot vector:

   - *Joint*: a joint semantic function / boundary tag including all the classes associated with a word in the input (e.g., "REQT[2]_M-¿FIGURE_REF_B").
   - *Hierarchical*: a single semantic function / boundary tag for each word in the input (e.g., "REQT[2]_M", "FIGURE_REF_B").

   The semantic functions are categorized into three levels, according to their dependency relationships:

   (a) CLAIM_NUM, INVENTION, REQT[#*Lv.*];

   (b) CLAIM_REF, FIG_REF;

(c) CLAIM_REF_NUM, FIG_REF.

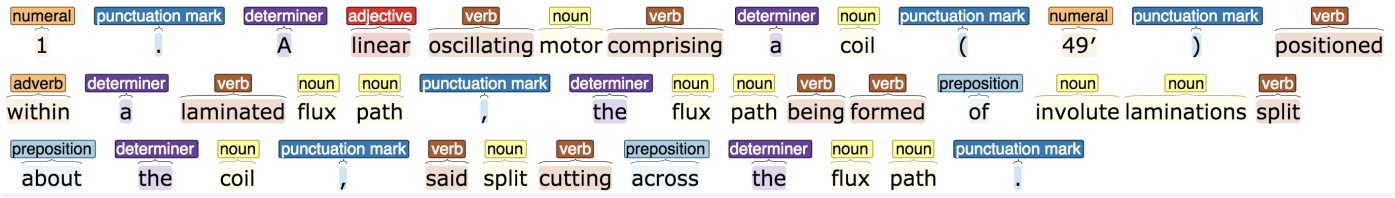
In the joint case, the levels are merged and tagged together with a single Neural Network. Otherwise, for each level, a separate instance of the Neural Network module is employed. The second and third levels take the one-hot encoded semantic function / boundary tag from the previous level as an additional input (see Figure 3.25).

An example of input/output pairs is provided for each module as follows:

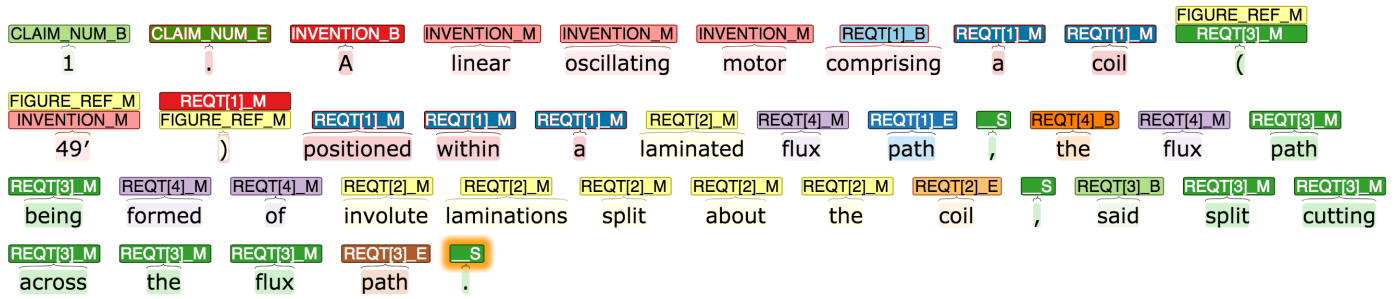
Claim sentence (input):
1 . A linear oscillating motor comprising a coil (49') positioned within a laminated flux path , the flux path being formed of involute laminations split about the coil , said split cutting across the flux path .

Morphological decomposition
1 . A linear oscillate -ing motor comprising a coil (49') position -ed within a laminate -ed flux path , the flux path be- ing form med of involute lamina -ation -s split about the coil , said split cut -ing a- cross the flux path .

POS-tagger



Claim segmentation



Figures 3.11, 3.24 and 3.25 illustrate the RNN module architectures.

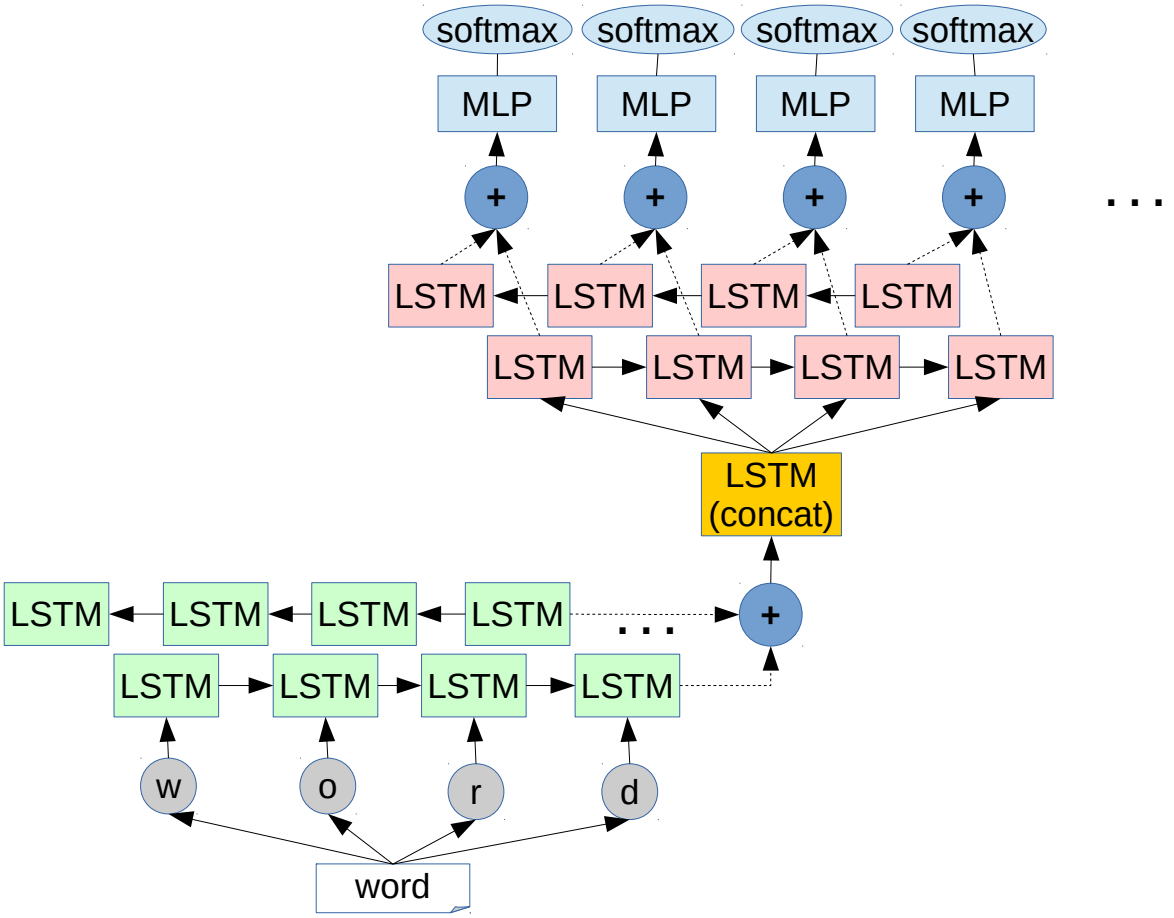


Figure 3.23: Sequence to sequence RNN architecture for morphological decomposition. The characters of the word to be decomposed are encoded as one-hot vectors and fed to the first Bi-LSTM as an input sequence. After all charachters have been processed, the final state of the encoder (the network's internal representation) is used as input for the decoder (the upper Bi-LSTM) followed by a Multi Layer Perceptron (MLP) and finally a softmax layer.

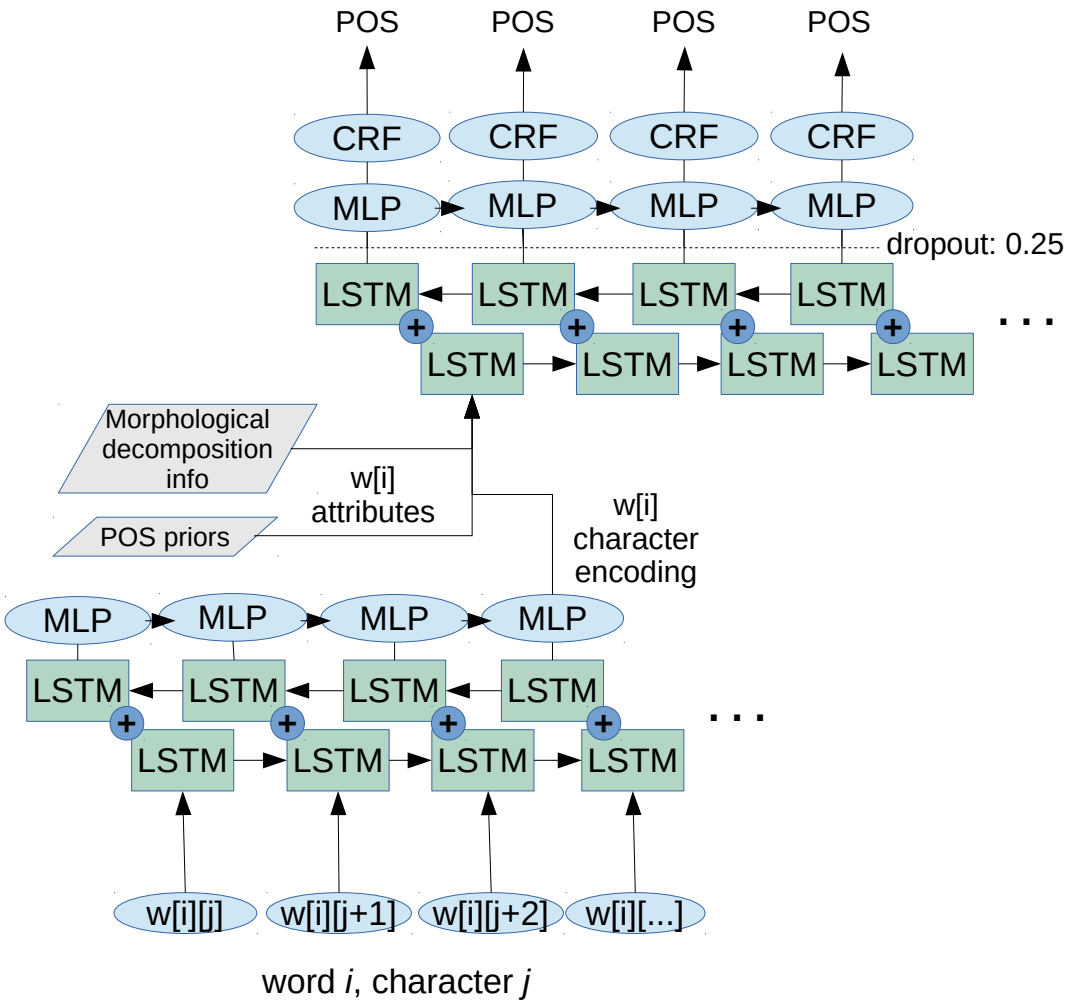


Figure 3.24: Bi-LSTM-CRF RNN architecture for POS-tagging. The characters of the word to be decomposed are encoded as one-hot vectors and fed to the first Bi-LSTM as an input sequence. If the word is morphologically decomposed, its surface form is substituted by a space-separated morpheme list. Additionally, POS classes for the word and each morpheme are passed as one-hot encoded input attributes.

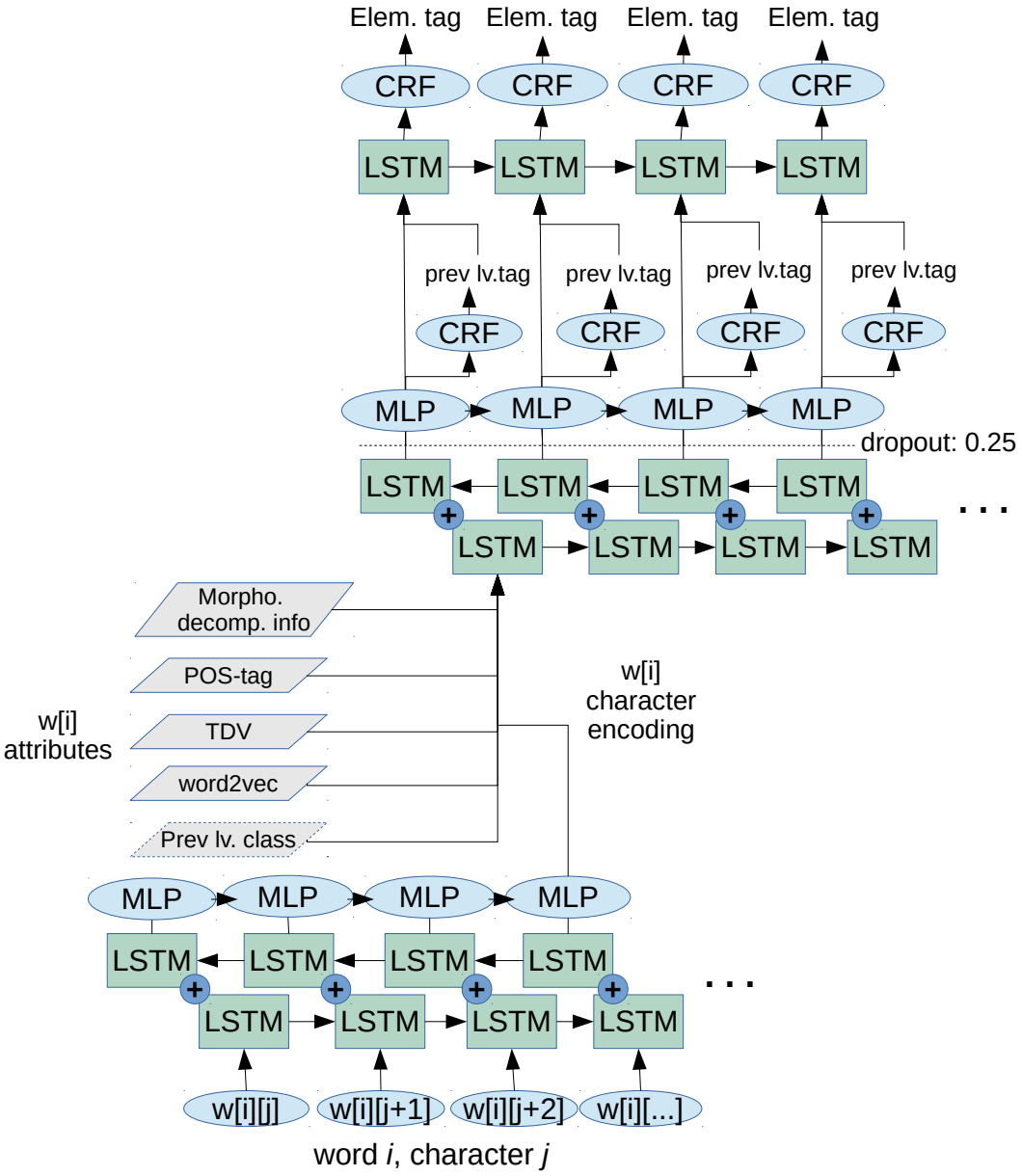


Figure 3.25: Bi-LSTM-CRF RNN architecture for claim segmentation. Additional to the inputs of the previous modules, it also receives word2vec embeddings and one-hot encoded TDV link types. The encoded class for the previous level semantic function is also used as input for 2nd and 3rd level taggers, when doing the hierarchical output.

Chapter 4

Experimental evaluation and discussion

Since the claim segmentation task is a novel contribution in this work, there is currently no other known methods to compare, nor public test sets for benchmarking. Therefore, to evaluate the segmentation performance of the whole system, a set of experiments was conducted with the reference annotation corpus produced with professional help (Section 3.5). As the goal of the segmentation method is to emulate the annotation process of a human expert, the chosen metrics and results analysis relate the success of the method to how close it matches the human made annotations.

Additionally, relevant comparisons are made to alternative methods of obtaining the linguistic features used by the segmentation method where appropriate.

4.1 Experimental setup and methodology

4.1.1 EPO data and reference annotation corpus

The European Patent Office (EPO) corpus was obtained from the CLEF-IP corpus [62], a test collection for patent classification and prior art retrieval published until 2011, and the largest structured corpus for patent information processing. It contains all EPO documents that have an application date previous to 2002 and also more than 400,000 World Intellectual Property Organization (WIPO) documents corresponding to European patent applications, totaling about 3 million documents and 1.5 million patents.

Due to the effort necessary to annotate all claims in each document, a small representative sample was needed to perform manual annotations that would scale to the entire collection. As presented in Section 3.5, this sampling was done in two stages:

1. Collect a random set of documents for each IPC subclass level symbol (649 symbols in total).

2. Collect a smaller subset of two documents for each topic, in a list of human expert chosen topics (10 topics in total)

An additional two extra documents (10% of the original smaller set) were randomly selected and added to the latter set, totaling 22 documents. Each claim in this set was manually annotated, resulting in close to 400 annotated claims, with an average length of

52 words, and standard deviation of 46 words. Those 22 annotated documents are set as the reference corpus for the experiments. The larger first set is to be used for posterior expansion of the reference corpus, through semi-automatic annotation using the current system. Manual annotation was done using the WebAnno tool [1] [63], with the structural view semantic functions described in Section 3.4 as the tag set (span annotations). The resulting annotation files can be found at `https://goo.gl/wnhKGW`.

4.1.2 Other data

For training the morphological decomposition module, the decomposition examples were collected from the Wiktionary corpus, as described in Section 3.8.3 and 3.8.4, with a total of about 220K word decompositions.

The training part of the *Universal Dependencies* English corpus [2] [61] was used for training the POS-tagger module, with a total of about 12.5K tagged sentences, 220K tagged words.

Word2vec [9] was trained using the entire EPO document corpus (about 44GB of plain text: abstracts, descriptions and claims), but including only one document per patent (the most recent one) and removing the documents from the reference corpus. An additional word2vec embedding file was obtained by training only the claims after morphological decomposition (about 6GB of plain text). It is used for obtaining TDV-W2V representations of OOV words.

4.1.3 Annotation modules setup

Each module: Morphological decomposition, POS-tagging and claim segmentation was set up and trained independently. Intermediate training data (encodings) and model statistics were stored as python native object files using SAF (Section 3.10.1), while trained ANN models where stored using *keras* [3] Neural Network library facilities, in HDF5 format files.

For testing, each module takes a list of SAF documents as input and outputs the same list with added annotations, corresponding to the RNN outputs. Additionally, module run statistics are included as follows:

- Morphological decomposition:
  - Softmax confidence per character or per word, depending on the settings.
  - Votes per solution, when using ensemble mode.
- POS-tagging:
  - POS priors (Wiktionary).
- Claim segmentation:
  - Per document precision and recall, with class weights.

[1] `https://www.ukp.tu-darmstadt.de/software/webanno/`
[2] `http://universaldependencies.org`
[3] https://keras.io/

The module outputs are either passed directly to the next one in the pipeline or stored for later use in case of repeated tests. All module annotations are optional, with the available ones being used as inputs from the second step of the pipeline.

Module parameters were set as follows:

- Morphological decomposition:
  - Maximum input size (chars): 40.
  - Maximum output size (chars): 68.
  - Context window size (chars): 3.
  - Encoder hidden layer size (LSTM): 400.
  - Decoder hidden layer sizes (LSTM, MLP): 300, 1000.
  - RNG distribution: truncated normal.
  - Ensemble size: 3.
  - Softmax confidence threshold: 0.8.
  - RNN training batch size: 200
  - RNN training epochs: 25
- POS-tagging:
  - Maximum input word size (chars): 45.
  - Maximum input size (words): 400.
  - Maximum input size (morphemes): 4.
  - Context window size (chars): 1.
  - Context window size (words): 5.
  - Character encoder hidden layer size (LSTM, MLP): 50, 200.
  - Hidden layer sizes (LSTM, MLP): 300, 800.
  - RNG distribution: uniform.
  - RNN training batch size: 50
  - RNN training epochs: 50
- Claim segmentation:
  - Maximum input word size (chars): 45.
  - Maximum training input size (words): 250.
  - Maximum testing input size (words): 400.
  - Maximum input size (morphemes): 4.
  - Context window size (chars): 1.
  - Context window size (words): 1, 3.
  - Number of TDV links: 10.
  - Word2vec dimension (d): 200.

- Word2vec parameters: cbow = 1, negative = 25, window = 10, iter = 15, sample = 1e-5.
- Character encoder hidden layer size (LSTM, MLP): 50, 200.
- Hidden layer sizes (LSTM, MLP): 300, 800.
- RNG distribution: uniform.
- RNN training batch size: 50 (20 for ctx. window size = 3)
- RNN training epochs: 10 (15 for ctx. window size = 3)

Parameters were taken from the original module training tasks, but not tuned in the case of morphological decomposition and POS-tagging. Parameter adjustment was done only for a single (first) fold in a leave-one-out test and left unchanged for the other ones.

4.1.4 Methodology

Given the small amount of annotated documents available, tests were done in a leave-one-out manner, to achieve complete coverage of topics and have statistical significance. This means that the claim segmentation module was trained with 21 documents and tested with the remaining one for each document in the reference corpus.

Metrics used for evaluation were defined per document as:

$$precision[class] = \frac{hits[class]}{test\_totals[class]} \tag{4.1}$$

$$recall[class] = \frac{hits[class]}{gold\_totals[class]} \tag{4.2}$$

$$weight[class] = \frac{gold\_totals[class]}{\sum_{\forall class} gold\_totals[class]} \tag{4.3}$$

$$precision = \frac{\sum_{\forall class} precision[class] * weight[class]}{\#classes} \tag{4.4}$$

$$recall = \frac{\sum_{\forall class} recall[class] * weight[class]}{\#classes} \tag{4.5}$$

$$F1\_score = 2 * \frac{precision * recall}{precision + recall} \tag{4.6}$$

where $hits[class]$ is the number of correct labels per class, $test\_totals[class]$ is the total number of tokens labeled as *class* by the system in the document, $gold\_totals[class]$ is the total number of tokens labeled as *class* in the reference corpus document (gold labels).

The metrics are weighted for per-class normalization. This is done so the classes with little presence on a document (REQT[>= 4]) do not over-penalize the score and the easy but always present classes (CLAIM_NUM, INVENTION) do not inflate the score. Per-segment metrics were not used due to the low amount of test data not making it statistically significant, though they will be included in future iterations of the system.

Overall precision, recall and F1-score is calculated by simply averaging the respective document measurements.

The leave-one-out test was run for the following attribute combinations:

- Char only: Only the character encoded tokens.
- POS: Char and Part-of-Speech.
- W2V: Char and word2vec embeddings.
- TDV_W2V: Char and Term Definition Vector ranked word2vec embeddings (Section 3.10.3).
- POS + W2V: Char, POS and word2vrc embeddings.
- POS + W2V + TDV_W2V: combination of all above.
- Morpho + POS: Surface form char and POS after morphological decomposition.
- Morpho + TDV_W2V: Surface form char and TDV links with OOV resolution.
- Morpho + POS + TDV_W2V: combination of the previous two.
- Morpho + POS + W2V + TDV_W2V: combination of all above.

Certain combinations were not tested due to having no theoretical benefit, e.g., Morpho + W2V, or just being detrimental to the system, e.g., Morpho + Char. The test was run more than 3 times to check for large score fluctuations, but none was found. Therefore the results presented are from the last run.

Additionally, for all attribute combinations including POS without morphological decomposition information, the same tests were done exchanging the POS-tagging module for the Stanford POS-tagger [4] [64]. This was done to compare the contribution of the morpheme-aware POS-tagger with a state-of-the-art system.

[4] `https://nlp.stanford.edu/software/tagger.shtml`

4.2 Results & Discussion

The experiment results are shown in Tables 4.1 and 4.2.

Table 4.1: Claim segmentation performance (joint model). Window size indicates the number of words used for obtaining attributes. A value of 3 means using left and right neighboring words.

Attribute combination	Window size	Precision	Recall	F1-score
Char only	1	0.12	0.11	0.11
Char only	3	0.15	0.20	0.17
POS	1	0.28	0.34	0.29
POS	3	0.33	0.40	0.35
POS (Stanford)	1	0.26	0.28	0.27
POS (Stanford)	3	0.35	0.38	0.36
W2V	1	0.65	0.61	0.62
W2V	3	0.66	0.60	0.62
TDV_W2V	1	0.63	0.57	0.59
TDV_W2V	3	0.65	0.62	0.63
POS + W2V	1	0.64	0.62	0.62
POS + W2V	3	0.63	0.62	0.62
POS (Stanford) + W2V	1	0.64	0.61	0.62
POS (Stanford) + W2V	3	0.63	0.62	0.62
POS + W2V + TDV_W2V	1	**0.67**	**0.63**	**0.64**
POS + W2V + TDV_W2V	3	0.64	0.61	0.62
POS (Stanford) + W2V + TDV_W2V	1	0.63	0.62	0.62
POS (Stanford) + W2V + TDV_W2V	3	0.65	0.63	0.64
Morpho + POS	1	0.27	0.34	0.29
Morpho + POS	3	0.36	0.40	0.37
Morpho + TDV_W2V	1	0.62	0.58	0.59
Morpho + TDV_W2V	3	0.64	0.62	0.62
Morpho + POS + TDV_W2V	1	0.62	0.56	0.58
Morpho + POS + TDV_W2V	3	0.62	0.59	0.60
Morpho + POS + W2V + TDV_W2V	1	**0.66**	**0.63**	**0.64**
Morpho + POS + W2V + TDV_W2V	3	0.64	0.62	0.63

Due to time and tooling constraints, only a part of the tests was done with the hierarchical model. The results are shown on Table 4.2.

Table 4.2: Claim segmentation performance (hierarchical model). Window size indicates the number of words used for obtaining attributes. A value of 3 means using left and right neighboring words.

Attribute combination	Window size	Precision	Recall	F1-score
Char only	1	0.38	0.27	0.30
Char only	3	0.40	0.32	0.34
POS	1	0.44	0.40	0.41
POS (Stanford)	1	0.41	0.42	0.41
W2V	1	0.74	0.71	0.72
TDV_W2V	1	0.72	0.65	0.68
Morpho + POS + W2V + TDV_W2V	1	**0.76**	**0.72**	**0.74**

The results indicate that the use of character-only encodings is not a viable option, given the current amount of training data available, this might change as with a significant increase in the number of training documents. Adding Part-of-Speech information

substantially increases performance and is quite positively affected by the context window size, something that was expected, as the claim element boundaries are mostly grammar-based. Word2vec embeddings provide the biggest leap in performance, indicating a strong favor to semantic discrimination in this task. The TDV_W2V ranked links provide the best balance, but can be further improved by the inclusion of POS information. The morphological decomposition information provides a slight increase in recall, at the cost of lowering precision in most cases.
The hierarchical tagging model provided a significant advantage over the joint one for the attribute combinations tested, emphasizing the role of semantic function dependencies in the performance.

Analysis of the OOV instances showed that in the reference corpus less than 70 words were OOV, including chemical compounds, from which 29 where successfully solved and represented, in a universe of approximately 1600 words. It is a statistically small (less than 2%), but relevant contribution to the task and further use of the generated semantic representations, as it allows better exploration of the annotations obtained in this work on future research.

The complementary contribution of TDV to the distributional-based representations provided by word2vec is indicated in the way the precision and recall are balanced for the former, although the tradeoff is still favorable as in the semantic similarity tests (Section 3.9.3).

Using the Stanford POS-tagger in place of the RNN POS-tagger module without morphemes causes the performance to decrease with window size = 1 and to increase when window size = 3. The performance of the RNN module is superior in both window sizes when using morphemes, but the difference is not significant after including the remaining attributes. This indicates that the Part-of-Speech contribution in the segmentation is subsumed by the W2V and TDV combination. While TDV links may explicitly define Part-of-Speech relations, the discriminative factor of word2vec seems to play an important part on this effect.

Another observation is that the increase in context window size does not help to increase performance overall, contrary to the expected behavior. Their performance fluctuations during training suggest that this is caused by a limit in the representation capacity of the RNN. In other words, the network may not have sufficient parameters to accommodate the large number of attributes. A solution is being investigated in terms of changing the dimension of hidden LSTM layers as a possible bottleneck. However, this implies other changes in the network architecture, such as adding perceptron layers and changing dropout rates, while keeping the number of parameters low enough so that the network can be trained under limited resources (Memory, GPU).

Although the current performance is still not sufficient for use of the system in a production environment, the outputs already allow reliable separation of claims regarding dependency, using the $CLAIM\_REF$ and $CLAIM\_REF\_NUM$ elements. Posterior extraction of a claims tree can be done by a simple accumulator and regular expression search on $CLAIM\_REF\_NUM$ elements, defining semantics for the "and", "or" and "any" operators (e.g., "*according to any previous claim*", "*as in claims 8 or 9*").

The identification of the invention and requirement elements is the main practical achievement, which can potentially increase productivity of a patent professional by a

large factor, as the manual separation of text in paper and spreadsheets for categorization is essentially skipped. At this stage of development, this can be leveraged by using the obtained segments in querying patent databases. An example would be automated querying for the contents of the *INVENTION* element in a document set, on a public patent office search system (e.g., EPO Espacenet [5], USPTO search [6]). This is such a task done manually by patent experts at the present, if the patent to be compared is new and hence not in a structured database.

Identification of requirement level, while the most lacking aspect in the tests (Section 4.3) facilitates alignment of the claim elements with IPC symbols and other classification systems used by patent offices. Finally, semantic comparison between said elements is also possible, by combining the element's Term Definition Vectors for each word or term. This is the basis of the subsequent experiments on prior art retrieval that are sought as the next step of this research.

Finally, the results indicate that the use of the collected linguistic features makes feasible the claim segmentation task, as proposed in this work. Considering the use of a Deep Learning-based method, such features allow effective segmentation even with a small amount of training data available. However, there is still a lot of room for improvement, some of which is discussed in the next section.

All the experimental code for the claim segmentation task can be obtained from the following public repositories:

- EPO data extraction: `https://github.com/nguyenlab/epo_extraction`
- Simple Annotation Framework: `https://github.com/nguyenlab/saf`
- Term Definition Vectors: `https://github.com/dscarvalho/tdv`
- Seq2Seq morphological decomposition: `https://github.com/nguyenlab/wikt_morphodecomp`
- TDV-enhanced Deep POS-tagger: `https://github.com/nguyenlab/wikt_deep_postagger`
- Claim segmentation tagger: `https://github.com/nguyenlab/claim_segmentation_tagger`

Experiment results data can be accessed at `https://goo.gl/xydrHk`.

4.3 Error analysis

The analysis of error cases in the experiments allows a better understanding of the system's flaws and how to correct them. The errors were divided in two categories: *a)* Annotation errors: are found in the outputs of the claim segmentation module, being the main type of mistake; *b)* Pre-processing errors: are introduced in the course of the annotation pipeline and possibly propagated by it. The latter type is usually the cause for the former, but may be just corrected internally if consistent, as part of the ANN function modeling.

[5] `https://worldwide.espacenet.com`

[6] `https://www.uspto.gov/patents-application-process/search-patents`

4.3.1 Annotation errors

The errors that caused the highest score penalties were in majority by wrong classification of the requirement level, as the nuances on changing levels are not obvious at first. Here is one example of such output, annotated as word [segment_boundary label]:

1 [CLAIM_NUM_B] . [CLAIM_NUM_E] A [INVENTION_B] cooled [INVENTION_M] panel [INVENTION_M] for [INVENTION_M] walls [INVENTION_M] of [INVENTION_M] electric [INVENTION_M] furnaces [INVENTION_E] , [__S] characterized [REQT[2]_B] by [REQT[2]_M] comprising [REQT[2]_M] ...

In this case, while the transition from the invention element to the first requirement was obvious due to the punctuation, the requirement was classified as a level 2, rather than the correct 1. As the entire segment got misclassified, and the REQT[1] class is highly weighted in the scores, this reasonably causes a high score penalty.

Possible solutions for this kind of mistake are: putting the requirement level as a separate output, so it is calculated as a separate function of the input, and increasing the amount of training data, in order to make such constraints evident.

Another common and impactful error case is the misclassification "in boundaries", as in the following example:

... A [INVENTION_B] linear [INVENTION_M] oscillating [REQT[2]_M] motor [INVENTION_M] comprising [REQT[1]_B] a [REQT[2]_M] coil [REQT[2]_M] ...

In this case, both the invention and first requirement elements have words that are classified with a different element label inside their boundaries. This is clearly a lack of discriminative power in the RNN function and should be solved with more training examples.

Additionally, certain requirement levels (6, 7) only appear in a single document, and thus are fated to be misclassified.

The remaining relevant annotation error instances are mostly boundary mistakes, which are typical of sequence tagging methods like the proposed one. Those are also likely to be corrected by the aforementioned solutions.

4.3.2 Errors introduced or propagated by the annotation pipeline

The only concerning error case found early in the pipeline was the apparently random POS mislabelling when applying morphological decomposition information, and using a window size greater than 1. The error was non-consistent regarding POS classes, but affected shorter words at a much greater rate then the longer ones, causing an unexpected large drop in performance in the Morpho + POS tests.

After analysis of the mislabellings, it was found that treating the morphemes as a fixed size list during the encoding of attributes was inserting noise in the inputs that was not detectable in the single input case (window size = 1) due to being inconsequential when combined with the POS priors. However, with a larger window size, the inputs became dependent of the noisy neighborhood specially on shorter words, for which there is little morpheme input support. This was solved by encoding the morpheme inputs as variable sized with zero-padding, thus obtaining more consistent inputs.

Of the error cases propagated in the pipeline, the most impactful is the POS mislabeling of punctuation marks, as they are critical to the separation of many claim elements.

Chapter 5

Conclusion

5.1 Summary & Concluding remarks

In this work, a method for identifying and classifying description elements in patent claims was developed, aiming to facilitate patent categorization and comparison. This was done by means of semantic analysis of the elements of invention described in claim text, opening the path for correctly isolating and producing semantically comparable representations of said elements. This methodology differs from current research in patent information extraction, in that it tries to emulate the initial step of a human expert process of claim analysis, separating claim descriptive elements, rather than using the document provided textual units.

The course of this work comprises a study on the several aspects of patent information processing, including the structuring of patent documents and claims and the computational representation of lexical and semantic aspects of natural language. Special attention was given to the linguistic characterization of each problem, which is reflected in the solutions developed at each information processing stage. Modeling of complex functions: morphological decomposition, Part-of-Speech tagging and the segmentation of claims, was approached with the use of Machine Learning methods appropriate for the problem in consideration, regarding fitness to the expected solutions and computational costs.

Major contributions of this work can be listed as follows:

a. The segmentation method for claim sentences, that emulates the initial annotation work done by patent experts, which is time consuming and relies on experience. They can then compare the claim elements by several criteria. Despite its benefits, no other research on this particular task was found at the moment.

b. The methods for gathering and exploiting multiple linguistic features from different sources, but especially Wiktionary. Those features enable effective use of Deep Learning even in cases where available training data is scarce.

Secondary contributions can be listed as follows, composing the overall system developed in the course of this research:

- A linguistically motivated language representation method, aimed at capturing and providing information unavailable through distributional approaches. This methods builds conceptual representations for lexical elements (morphemes, words, phrases)

using information extracted from a collaborative language resource (Wiktionary). The representations obtained by this method take the form of sparse vectors called *Term Definition Vectors* (TDV), which are human interpretable due to its dimensions being mapped to documented types.

- An efficient approach to patent document segmentation, through the use of a computationally inexpensive method of combining distributed semantic representations (word embeddings) into sentence representations. The method for combining word embeddings, called *Term Order Probabilities* (TOP) is based on a linguistic premise of syntactic regularity to use a statistical "shortcut" to obtain useful sentence embeddings.

- A morphological decomposition system using softmax function values as sequence-wide confidence measures for pruning outputs. Such system was made viable by the development of a method to collect morphological decomposition examples from etymology descriptions found on Wiktionary.

All contributions here presented performed well in the relevant experiments, having competitive or state-of-the art performance where applicable.

The claim segmentation experimental results indicate the feasibility of the claim segmentation task as proposed in this work, with the use of the collected linguistic features. Such features allow effective segmentation using a Deep Learning-based method, even with a relatively small amount of training data available. The annotation principles used for the segmentation guarantee the usefulness of the results to the patent expert community, as they get improved. There is, however, still a lot of room for improvement.

The processing of information from patent documents still present a big set of challenges for years to come and this work intends to open new avenues of exploration. Cooperation between NLP researchers and patent professionals is something seem as an important element in future developments in this field.

5.2 Directions for future research

Experimental evaluation of the semantic segmentation model should be extended, and more annotation material for a reference corpus on claim segmentation is currently being produced. Other immediate work include a semi-supervised learning test over a controlled document sample to evaluate the expressiveness of the semantic representations obtained from the claim elements and also to initiate an automatic annotation effort covering all the EPO and WIPO patent documents.

With the availability of accurate claim segment representations comes the possibility to use then for prior art retrieval and classification, which are the next goals of this work. Other future research include the development of a model for extracting semantic relations between claim elements, and to extend the current claim analysis method to other types of legal text.

Bibliography

[1] S. Sheremetyeva: "Automatic Text Simplification For Handling Intellectual Property (The Case of Multiple Patent Claims)", COLING 2014, p. 41 (2014).

[2] S. Brügmann et al.: "Towards content-oriented patent document processing: Intelligent patent analysis and summarization", World Patent Information, vol. 40, pp. 30 – 42 (2015), ISSN 0172-2190, doi:http://dx.doi.org/10.1016/j.wpi.2014.10.003, URL http://www.sciencedirect.com/science/article/pii/S0172219014001410.

[3] S.-H. Huang, H.-R. Ke and W.-P. Yang: "Structure clustering for Chinese patent documents", Expert Systems with Applications, vol. 34, no. 4, pp. 2290 – 2297 (2008), ISSN 0957-4174, doi:http://dx.doi.org/10.1016/j.eswa.2007.03.012, URL http://www.sciencedirect.com/science/article/pii/S0957417407001224.

[4] Q. V. Le and T. Mikolov: "Distributed Representations of Sentences and Documents.", in ICML, vol. 14, pp. 1188–1196 (2014).

[5] H. Yin, C. Zhang, Y. Zhu and Y. Ji: "Representing sentence with unfolding recursive autoencoders and dynamic average pooling", in Data Science and Advanced Analytics (DSAA), 2014 International Conference on, pp. 413–419, IEEE (2014).

[6] D. S. Carvalho and M. L. Nguyen: "Efficient Neural-based patent document segmentation with Term Order Probabilities", in Proceedings of the 25th European Symposium on Artificial Neural Networks, ESANN 2017, Université catholique de Louvain (April 2017), URL https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2017-123.pdf.

[7] J. D. Lafferty, A. McCallum and F. C. N. Pereira: "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", in Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01, pp. 282–289, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001), ISBN 1-55860-778-1, URL http://dl.acm.org/citation.cfm?id=645530.655813.

[8] M. Sahlgren: "The distributional hypothesis", Italian Journal of Linguistics, vol. 20, no. 1, pp. 33–54 (2008).

[9] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean: "Distributed representations of words and phrases and their compositionality", Advances in neural information processing systems (2013).

[10] J. Pennington, R. Socher and C. Manning: "Glove: Global Vectors for Word Representation", in Proceedings of the 2014 Conference on Empirical Methods in Natural

Language Processing (EMNLP), pp. 1532–1543, Association for Computational Linguistics (2014), doi:10.3115/v1/D14-1162, URL http://aclweb.org/anthology/D14-1162.

[11] O. Levy and Y. Goldberg: "Dependency-Based Word Embeddings", in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 302–308, Association for Computational Linguistics (2014), doi:10.3115/v1/P14-2050, URL http://aclweb.org/anthology/P14-2050.

[12] W. S. McCulloch and W. Pitts: "A logical calculus of the ideas immanent in nervous activity", The bulletin of mathematical biophysics, vol. 5, no. 4, pp. 115–133 (1943).

[13] F. Rosenblatt: "Principles of neurodynamics", (1962).

[14] I. Aleksander and H. Morton: An introduction to neural computing, vol. 3, Chapman & Hall London (1990).

[15] P. D. Wasserman: Neural computing: theory and practice, Van Nostrand Reinhold Co. (1989).

[16] M. Collins: "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms", in Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10, pp. 1–8, ACL (2002).

[17] USPTO: "Google USPTO patents. (https://www.google.com/googlebooks/uspto-patents.html)", (2017), URL https://www.google.com/googlebooks/uspto-patents.html, [Online; accessed 04-December-2017].

[18] L. Buitinck: "seqlearn: Sequence learning toolkit for Python", (2017), URL https://github.com/larsmans/seqlearn, [Online; accessed 04-December-2017].

[19] N. Ghoula, K. Khelif and R. Dieng-Kuntz: "Supporting patent mining by using ontology-based semantic annotations", in Web intelligence, IEEE/WIC/ACM international conference on, pp. 435–438, IEEE (2007).

[20] S. Taduri, G. T. Lau, K. H. Law and J. P. Kesan: "A patent system ontology for facilitating retrieval of patent related information", in Proceedings of the 6th International Conference on Theory and Practice of Electronic Governance, pp. 146–157, ACM (2012).

[21] S.-Y. Yang and V.-W. Soo: "Extract conceptual graphs from plain texts in patent claims", Engineering Applications of Artificial Intelligence, vol. 25, no. 4, pp. 874–887 (2012).

[22] B. T. Hung, N. Le Minh and A. Shimazu: "Divide and Translate Legal Text Sentence by Using Its Logical Structure", in Knowledge, Information and Creativity Support Systems (KICSS), 2012 Seventh International Conference on, pp. 18–23, IEEE (2012).

[23] R. Girju, A. Badulescu and D. Moldovan: "Automatic discovery of part-whole relations", Computational Linguistics, vol. 32, no. 1, pp. 83–135 (2006).

[24] P. Pantel and M. Pennacchiotti: "Espresso: Leveraging generic patterns for automatically harvesting semantic relations", in Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, pp. 113–120, Association for Computational Linguistics (2006).

[25] M. Le Nguyen and A. Shimazu: "A semi supervised learning model for mapping sentences to logical forms with ambiguous supervision", Data & Knowledge Engineering, vol. 90, pp. 1–12 (2014).

[26] T. Roh, Y. Jeong and B. Yoon: "Developing a Methodology of Structuring and Layering Technological Information in Patent Documents through Natural Language Processing", Sustainability, vol. 9, no. 11 (2017), ISSN 2071-1050, doi: 10.3390/su9112117, URL http://www.mdpi.com/2071-1050/9/11/2117.

[27] I. Sutskever, O. Vinyals and Q. V. Le: "Sequence to sequence learning with neural networks", in Advances in neural information processing systems, pp. 3104–3112 (2014).

[28] World Intellectual Property Organization: "International Patent Classification (IPC)", (2017), URL http://www.wipo.int/classifications/ipc/en/.

[29] M. Minsky and S. Papert: "Perceptrons.", Cambridge, Ma (1969).

[30] C. Cortes and V. Vapnik: "Support-vector networks", Machine learning, vol. 20, no. 3, pp. 273–297 (1995).

[31] R. Raina, A. Madhavan and A. Y. Ng: "Large-scale Deep Unsupervised Learning Using Graphics Processors", in Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, pp. 873–880, ACM, New York, NY, USA (2009), ISBN 978-1-60558-516-1, doi:10.1145/1553374.1553486, URL http://doi.acm.org/10.1145/1553374.1553486.

[32] S. Hochreiter and J. Schmidhuber: "Long Short-Term Memory", Neural Computation, vol. 9, no. 8, pp. 1735–1780 (1997), doi:10.1162/neco.1997.9.8.1735, URL https://doi.org/10.1162/neco.1997.9.8.1735.

[33] J. H. Greenberg: "A quantitative approach to the morphological typology of language", International journal of American linguistics, vol. 26, no. 3, pp. 178–194 (1960).

[34] G. Libben, M. Gibson, Y. B. Yoon and D. Sandra: "Compound fracture: The role of semantic transparency and morphological headedness", Brain and Language, vol. 84, no. 1, pp. 50 – 64 (2003), ISSN 0093-934X, doi:https://doi.org/10.1016/S0093-934X(02)00520-5, URL http://www.sciencedirect.com/science/article/pii/S0093934X02005205, brain and Language Special Issue.

[35] A. Pirkola: "Morphological typology of languages for IR", Journal of Documentation, vol. 57, no. 3, pp. 330–348 (2001), doi:10.1108/EUM0000000007085, URL https://doi.org/10.1108/EUM0000000007085.

[36] R. H. Baud, A.-M. Rassinoux, P. Ruch, C. Lovis and J.-R. Scherrer: "The power and limits of a rule-based morpho-semantic parser.", in Proceedings of the AMIA symposium, p. 22, American Medical Informatics Association (1999).

[37] G. Thurmair: "Comparing rule-based and statistical MT output", in The Workshop Programme, p. 5 (2004).

[38] M. Konkol and M. Konopík: "Maximum entropy named entity recognition for czech language", in Text, Speech and Dialogue, pp. 203–210, Springer (2011).

[39] H. C. Carneiro, F. M. França and P. M. Lima: "Multilingual part-of-speech tagging with weightless neural networks", Neural Networks, vol. 66, pp. 11–21 (2015).

[40] T. Sakakini, S. Bhat and P. Viswanath: "MORSE: Semantic-ally Drive-n MORpheme SEgment-er", in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 552–561, ACL (2017).

[41] M. Faruqui and C. Dyer: "Non-distributional Word Vector Representations", in Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pp. 464–469, Association for Computational Linguistics (2015), doi:10.3115/v1/P15-2076, URL `http://aclweb.org/anthology/P15-2076`.

[42] M. Kurimo, K. Lagus, S. Virpioja and V. Turunen: "Morpho Challenge", (2010), URL `http://morpho.aalto.fi/events/morphochallenge/`, [Online; accessed 04-December-2017].

[43] Wikimedia Foundation: "Wiktionary - The free dictionary", (2017), URL `https://www.wiktionary.org`, [Online; accessed 04-December-2017].

[44] D. S. Carvalho, M. L. Nguyen and H. Iida: "'An Analysis of Majority Voting in Homogeneous Groups for Checkers: Understanding Group Performance through Unbalance", in Advances in Computer Games 2017, Springer (April 2017), URL `https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2017-123.pdf`.

[45] O. Kohonen, S. Virpioja, L. Leppänen and K. Lagus: "Semi-supervised extensions to morfessor baseline", in Proceedings of the Morpho Challenge 2010 Workshop, pp. 30–34 (2010).

[46] S. Virpioja, P. Smit, S.-A. Grönroos, M. Kurimo et al.: "Morfessor 2.0: Python implementation and extensions for Morfessor Baseline", Tech. rep. (2013).

[47] M. T. Pilehvar and R. Navigli: "From senses to texts: An all-in-one graph-based approach for measuring semantic similarity", Artificial Intelligence, vol. 228, pp. 95–128 (2015).

[48] J. Derrac and S. Schockaert: "Inducing semantic relations from conceptual spaces: a data-driven approach to plausible reasoning", Artificial Intelligence, vol. 228, pp. 66–94 (2015).

[49] C. Fellbaum et al.: "WordNet: An electronic database", (1998).

[50] T. H. Haveliwala: "Topic-sensitive pagerank", in Proceedings of the 11th international conference on World Wide Web, pp. 517–526, ACM (2002).

[51] C. K. Ogden, I. A. Richards, S. Ranulf and E. Cassirer: "The Meaning of Meaning. A Study of the Influence of Language upon Thought and of the Science of Symbolism", (1923).

[52] D. S. Carvalho and M. L. Nguyen: "Building Lexical Vector Representations from Concept Definitions", in Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017) (preprint), Association for Computational Linguistics (2017), URL `http://www.aclweb.org/anthology/E/E17/E17-1085.pdf`.

[53] D. S. Carvalho and M. L. Nguyen: "WikTDV: Data extraction and vector representation resource for Wiktionary senses", in Proceedings of the 9th International Conference on Knowledge and Systems Engineering, KSE 2017, IEEE (October 2017), URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8119465`.

[54] F. Hill, R. Reichart and A. Korhonen: "SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation", Computational Linguistics, vol. 41, no. 4, pp. 665–695 (2015), doi:10.1162/COLI_a_00237, URL `http://aclweb.org/anthology/J15-4004`.

[55] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman and E. Ruppin: "Placing search in context: The concept revisited", in Proceedings of the 10th international conference on World Wide Web, pp. 406–414, ACM (2001).

[56] H. Rubenstein and J. B. Goodenough: "Contextual correlates of synonymy", Communications of the ACM, vol. 8, no. 10, pp. 627–633 (1965).

[57] E. Bruni, N.-K. Tran and M. Baroni: "Multimodal Distributional Semantics", Journal of Artificial Intelligence Research (JAIR), vol. 49, no. 1-47 (2014).

[58] A. K. Nguyen, S. Schulte im Walde and T. N. Vu: "Integrating Distributional Lexical Contrast into Word Embeddings for Antonym-Synonym Distinction", in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 454–459, Association for Computational Linguistics (2016), doi:10.18653/v1/P16-2074, URL `http://aclweb.org/anthology/P16-2074`.

[59] G. Recski, E. Iklódi, K. Pajkossy and A. Kornai: Proceedings of the 1st Workshop on Representation Learning for NLP, chap. Measuring Semantic Similarity of Words Using Concept Networks, pp. 193–200, Association for Computational Linguistics (2016), doi:10.18653/v1/W16-1622.

[60] T. Joachims: "Training linear SVMs in linear time", in Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 217–226, ACM (2006).

[61] S. Petrov, D. Das and R. McDonald: "A universal part-of-speech tagset", In Proceedings of LREC (2011).

[62] Information Management and Preservation, IFS - Vienna University of Technology: "CLEF-IP Corpus", (2017), URL http://www.ifs.tuwien.ac.at/~clef-ip/download-central.shtml, [Online; accessed 04-December-2017].

[63] C. Biemann, K. Bontcheva, R. E. de Castilho, I. Gurevych and S. M. Yimam: "Collaborative Web-based Tools for Multi-layer Text Annotation", The Handbook of Linguistic Annotation', Text, Speech, and Technology book series, Springer Netherlands (2017).

[64] K. Toutanova, D. Klein, C. D. Manning and Y. Singer: "Feature-rich part-of-speech tagging with a cyclic dependency network", in Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1, pp. 173–180, Association for Computational Linguistics (2003).

[65] World Intellectual Property Organization: "WIPO IP Statistics Data Center", (2017), URL https://www3.wipo.int/ipstats/index.htm?tab=patent.

Publications

[1] D.S. Carvalho, M.T. Nguyen, C.X. Tran, M.L. Nguyen: "Lexical-Morphological Modeling for Legal Text Analysis," New Frontiers in Artificial Intelligence (Lecture Notes in Artificial Intelligence), pp.295-311 (Jan. 2016). DOI: 10.1007/978-3-319-50953-2

[2] D.S. Carvalho, V.D. Tran, K.V. Tran, V.D. Lai, M.L. Nguyen: "Lexical to Discourse-level Corpus Modeling for Legal Question Answering," In Proceedings of the 10th International Workshop on Juris-informatics, JURISIN 2016 (Nov. 2016).

[3] D.S. Carvalho and M.L. Nguyen: "Building Lexical Vector Representations from Concept Definitions," In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 (Apr. 2017).

[4] D.S. Carvalho and M.L. Nguyen: "Efficient Neural-based patent document segmentation with Term Order Probabilities," In Proceedings of the 25th European Symposium on Artificial Neural Networks, ESANN 2017 (Apr. 2017).

[5] D.S. Carvalho, V.D. Tran, K.V. Tran, M.L. Nguyen: "Improving Legal Information Retrieval by Distributional Composition with Term Order Probabilities," In Proceedings of Competition on Legal Information Extraction/Entailment (COLIEE 2017), ICAIL 2017 (Jun. 2017).

[6] D.S. Carvalho, M.L. Nguyen, H. Iida: "An Analysis of Majority Voting in Homogeneous Groups for Checkers: Understanding Group Performance through Unbalance," Advances in Computer Games 2017 (Jul. 2017).

[7] D.S. Carvalho and M.L. Nguyen: "WikTDV: Data extraction and vector representation resource for Wiktionary senses," In Proceedings of the 9th International Conference on Knowledge and Systems Engineering, KSE 2017 (Oct. 2017).

Appendices

Appendix A

Complementary figures

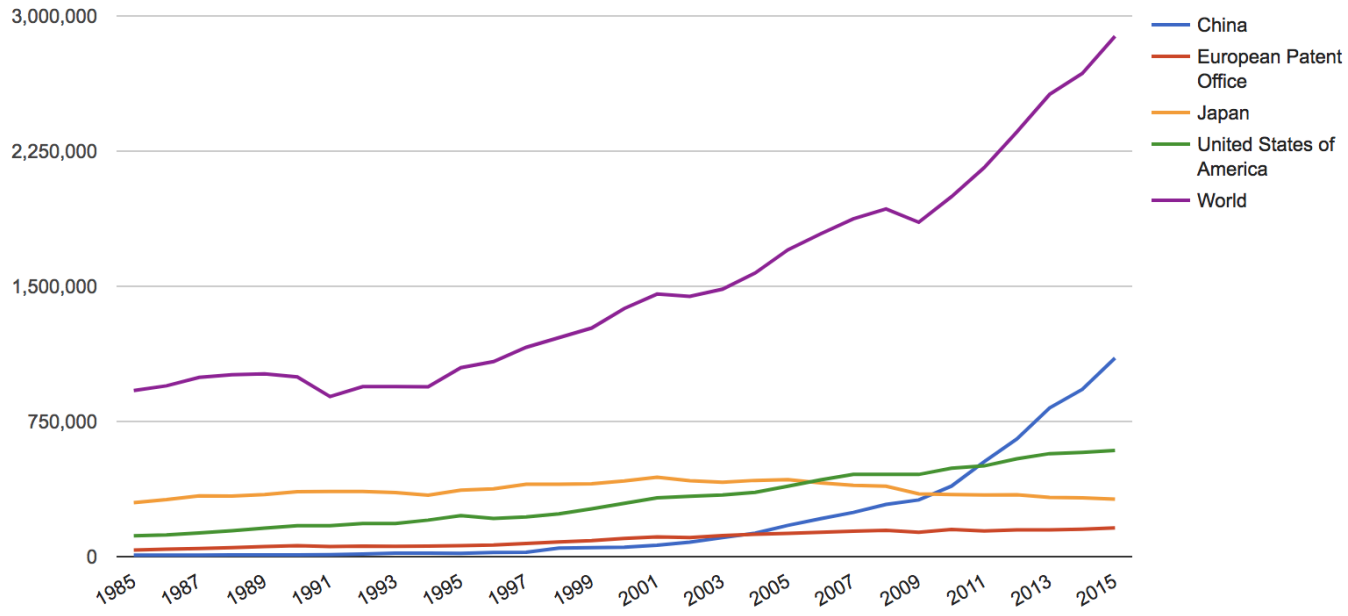


Figure A.1: Total patent applications per year, in the period 1985-2015. (WIPO 2017 [65])

Appendix B

Claims example: WO0074470 (A1)

CLAIMS

1. Plant seeds that contain an oil having an oleic acid content of more than 40 wt% and a stearic acid content of more than 12 wt% based on the total fatty acid content of said oil, and wherein a maximum of 10 wt% of the fatty acid groups in the sn-2 position of the TAG molecules constituting the oil are saturated fatty acid groups.

2. Plant seeds according to claim 1, wherein the seeds contain an oil that has in the sn-2 position of the TAG molecules constituting the oil a maximum of 8 wt% of saturated fatty acid groups.

3. Plant seeds according to claims 1 or 2, wherein the seeds contain an oil that has in the sn-2 position of the TAG molecules constituting the oil a maximum of 5 wt% of saturated fatty acid groups.

4. Plant seeds according to claims 1-3, wherein the oleic acid content is from 55 to 75 wt%.

5. Plant seeds according to claims 1-4, wherein the stearic acid content is from 15 to 50 wt%.

6. Plant seeds according to claim 5, wherein the stearic acid content is from 20 to 40 wt%.

7. Plant seeds according to claims 1-6, wherein the oil has a total level of saturated fatty acids of at least 20 wt%.

8. Plant seeds according to claims 1-7, wherein the oil has a linoleic acid content of less than 20 wt%.

9. Plant seeds according to claims 1-8, characterized in that said seeds are sunflower seeds.

10. Oil having an oleic acid content of more than 40 wt% and a stearic acid content of more than 12 wt% based on the total fatty acid content of said oil, and wherein a maximum of 10 wt% of the fatty acid groups in the sn-2 position of the TAG molecules constituting the oil are saturated fatty acid groups.

11. Oil as claimed in claim 10, as contained in plant seeds as claimed in claims 1-9.

12. Plants grown from plant seeds according to claims 1-9.

13. Plants producing plant seeds according to claims 1-9.

14. Method for producing a plant which forms seeds as claimed in claims 1-9, which method comprises:

a) providing seeds which contain an oil having a stearic acid content of at least 12 wt%;

b) providing seeds which contain an oil having an oleic acid content of at least 40 wt% and a thioesterase activity over stearoyl-ACP of at least 10% of the thioesterase activity over oleoyl-ACP;

c) crossing plants grown from the seeds provided in step (a) and (b);

d) harvesting the F1 seed progeny.

15. Method as claimed in claim 14, further comprising the steps of:

e) planting the F1 progeny seeds to grow plants;

f) self-pollinating the plants thus grown to produce F2 seed;

g) testing the seed for the presence of a stearic acid content of at least 12 wt%, an oleic acid content of at least 40 wt% and a thioesterase activity over stearoyl-ACP of at least 10% of the thioesterase activity over oleoyl-ACP;

h) planting seeds having the desired levels of stearic acid content, oleic acid content and thioesterase activity to grow plants;

i) self-pollinating the plants thus grown to produce F3 seed; and

j) optionally repeating steps g), h) and i) until the desired levels of stearic acid content and oleic acid content and the high thioesterase activity are fixed.

16. Method as claimed in claims 14 and 15, wherein the seeds which contain an oil having a stearic acid content of at least 12 wt% are provided by:

a) mutagenic treatment of seeds having a stearic acid content of less than 12%;

b) producing plants therefrom which are pollinated to produce seeds;

c) testing the seeds for the desired stearic acid content; and

d) optionally repeating steps b) and c).

17. Method as claimed in claims 14-16, wherein the seeds are sunflower seeds.

18. Meal or crushed seeds originating from seeds according to claims 1-9.

Appendix C

Claims analysis example: EP0915965 (B1), Claim 13

Claim

A method of fabricating a bioreactor of any of claims 1 to 7, the method comprising: a) providing a hollow filament bioreactor cartridge, the cartridge comprising a housing containing a plurality of elongate hollow filaments each positioned within the housing substantially parallel to the central axis and defining an extrafilamentary space within the housing, each of the hollow filaments formed of a material which allows molecular transport therethrough, the housing further comprising a filament inlet port and a filament outlet port, said ports communicating through the hollow filaments to define a filament flow path, and a housing inlet port and a housing outlet port, said ports communicating through the extrafilamentary space to define an extrafilament flow path, the extrafilament flow path being isolated from the filament flow path such that a material in one path may enter the other path only by molecular transport through the material comprising the hollow filaments; and b) introducing a volume of a gellable material into the housing in a manner such that it becomes positioned in the region of the housing at which the housing outlet port is located or a short distance into the extrafilamentary space toward the housing inlet port, the volume of the gellable material such that, upon gelling, the resulting gel will form a hydrogel plug positioned in the extrafilament flow path to maintain a uniform flow across the extrafilament flow path.

Claim elements

#	Element	Analysis
01	A method of fabricating a bioreactor	The invention core matter.
02	of any of claims 1 to 7	First level requirement, specifies dependence on referred claims.
03	the method comprising: a) providing a hollow filament bioreactor cartridge	2nd level requirement, details composition of bioreactor given by (02) claims.
04	the cartridge comprising a housing containing a plurality of elongate hollow filaments	3rd level requirement, details (03) .
05	each positioned within the housing substantially parallel to the central axis and defining an extrafilamentary space within the housing	4th level requirement, detailing hollow filaments from (04).
06	each of the hollow filaments formed of a material which allows molecular transport therethrough	4th level requirement, also detailing hollow filaments from (04).
07	the housing further comprising a filament inlet port and a filament outlet port	4th level requirement, detailing (04) housing.
08	said ports communicating through the hollow filaments to define a filament flow path	5th level requirement, detailing (07) ports.
09	and a housing inlet port and a housing outlet port	4th level requirement, also detailing (04) housing.
10	said ports communicating through the extrafilamentary space to define an extrafilament flow path	5th level requirement, detailing (09) ports.
11	the extrafilament flow path being isolated from the filament flow path such that a material in one path may enter the other path only by molecular transport through the material comprising the hollow filaments	6th level requirement, detailing the extrafilament flow path from (10).
12	and b) introducing a volume of a gellable material into the housing in a manner such that it becomes positioned in the region of the housing at which the housing outlet port is located or a short distance into the extrafilamentary space toward the housing inlet port	2nd level requirement, continues detailing the composition of bioreactor given by (02) claims.
13	the volume of the gellable material such that, upon gelling, the resulting gel will form a hydrogel plug positioned in the extrafilament flow path to maintain a uniform flow across the extrafilament flow path	3rd level requirement, detailing the volume of the gellable material from (12).