ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ
http://www.eee.uniwa.gr
http://www.idpe.uniwa.gr
Θηβών 250, Αθήνα-Αιγάλεω 12241
Τηλ: +30 210 538-1614
Διατμηματικό Πρόγραμμα Μεταπτυχιακών Σπουδών
Τεχνητή Νοημοσύνη και Βαθιά Μάθηση
https://aidl.uniwa.gr/

UNIVERSITY OF WEST ATTICA
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING
DEPARTMENT OF INDUSTRIAL DESIGN AND PRODUCTION ENGINEERING
http://www.eee.uniwa.gr
http://www.idpe.uniwa.gr
250, Thivon Str., Athens, GR-12241, Greece
Tel: +30 210 538-1614
Master of Science in
Artificial Intelligence and Deep Learning
https://aidl.uniwa.gr/

Master of Science Thesis

Greek Patent Classification Using Deep Learning

Student: Pontikis Ioannis
Registration Number: AIDL-0012

MSc Thesis Supervisor

Dr. Kasnesis Panagiotis

ATHENS-EGALEO, FEBRUARY 2023

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ
http://www.eee.uniwa.gr
http://www.idpe.uniwa.gr
Θηβών 250, Αθήνα-Αιγάλεω 12241
Τηλ: +30 210 538-1614
Διατμηματικό Πρόγραμμα Μεταπτυχιακών Σπουδών
Τεχνητή Νοημοσύνη και Βαθιά Μάθηση
https://aidl.uniwa.gr/

UNIVERSITY OF WEST ATTICA
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING
DEPARTMENT OF INDUSTRIAL DESIGN AND PRODUCTION ENGINEERING
http://www.eee.uniwa.gr
http://www.idpe.uniwa.gr
250, Thivon Str., Athens, GR-12241, Greece
Tel: +30 210 538-1614
Master of Science in
Artificial Intelligence and Deep Learning
https://aidl.uniwa.gr/

Μεταπτυχιακή Διπλωματική Εργασία

Ταξινόμηση Πατεντών με την Χρήση Βαθιάς Μάθησης

Φοιτητής: Ποντίκης Ιωάννης
ΑΜ: AIDL-0012

Επιβλέπων Καθηγητής

Δρ. Παναγιώτης Κασνέσης

ΑΘΗΝΑ-ΑΙΓΑΛΕΩ, ΦΕΒΡΟΥΑΡΙΟΣ 2023

This MSc Thesis has been accepted, evaluated, and graded by the following committee:

Supervisor	Member	Member
Panagiotis Kasnesis Digitally signed by Panagiotis Kasnesis Date: 2023.02.22 23:26:28 +02'00'	Eleni Aikaterini Leligkou Digitally signed by Eleni Aikaterini Leligkou Date: 2023.02.28 16:12:55 +02'00'	Periklis Papadopoulos Digitally signed by Periklis Papadopoulos Date: 2023.02.28 19:30:11 +02'00'
Kasnesis Panagiotis	Helen C. Leligou	Papadopoulos Perikles
Lecturer	Associate Professor	Professor
Electrical & Electronics Engineering	Industrial Design & Production Engineering	Electrical & Electronics Engineering
University of West Attica	University of West Attica	University of West Attica

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο/η κάτωθι υπογεγραμμένος Ιωάννης Ποντίκης του Αντωνίου, με αριθμό μητρώου AIDL-0012 μεταπτυχιακός φοιτητή του ΔΠΜΣ «Τεχνητή Νοημοσύνη και Βαθιά Μάθηση» του Τμήματος Ηλεκτρολόγων και Ηλεκτρονικών Μηχανικών και του Τμήματος Μηχανικών Βιομηχανικής Σχεδίασης και Παραγωγής, της Σχολής Μηχανικών του Πανεπιστημίου Δυτικής Αττικής,

δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της μεταπτυχιακής διπλωματικής εργασίας και κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος. Η εργασία δεν έχει κατατεθεί στο πλαίσιο των απαιτήσεων για τη λήψη άλλου τίτλου σπουδών ή επαγγελματικής πιστοποίησης πλην του παρόντος.
Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του διπλώματός μου.»

Ο Δηλών

Ιωάννης Ποντίκης

[illegible]

(Υπογραφή φοιτητή)

Declaration of the author of this MSc thesis

I, Ioannis Pontikis, Antonios with the following student registration number: AIDL-0012, postgraduate student of the MSc programme in "Artificial Intelligence and Deep Learning", which is organized by the Department of Electrical and Electronic Engineering and the Department of Industrial Design and Production Engineering of the Faculty of Engineering of the University of West Attica, hereby declare that:

I am the author of this MSc thesis and any help I may have received is clearly mentioned in the thesis. Additionally, all the sources I have used (e.g., to extract data, ideas, words, or phrases) are cited with full reference to the corresponding authors, the publishing house, or the journal; this also applies to the Internet sources that I have used. I also confirm that I have personally authored this thesis and the intellectual property rights belong to myself and to the University of West Attica. This work has not been submitted for any other degree or professional qualification except as specified in it.

Any violations of my academic responsibilities, as stated above, constitutes substantial reason for the cancellation of the conferred MSc degree.

The author

Ioannis Pontikis

[illegible]

(Signature)

Acknowledgements

I would like to express my deepest appreciation to Dr. Panagiotis Kasnesis which supervised and supported this project as much as he could. Also, his help in the research was significantly useful by providing tools and solutions to each step. He also made possible the publication of this project to NIDS2022.

I would also like to express my sincere gratitude to my manager Dr. Eustratios Koutivas which also provided all the useful resources and tools for this project. Also, his help in problem solving and getting the right mindset was beyond any expectation. He is a great researcher and a great manager and also, he is a person that made this project possible by trusting me in having this master and also this project.

Finally, I would like to help the Hellenic Industrial Property Organization and the University of West Attica for this project since both contribute to making this project possible with their infrastructures.

Abstract

Patents are documents that contain state of the art technical and scientific information in almost every field of science. Patent applications filed every day in the Hellenic Industrial Property Organization, have to be intellectually classified by domain experts based on a hierarchical taxonomy. As a result, this classification process is labor intensive and overwhelming for the experts. This paper proposes the use of Natural Language Processing (NLP) for automated patent classification. We compare state of the art deep learning-based NLP methods for the automated semantic categorization of these documents. For this purpose, we built a dataset comprised by around 70,000 Greek patents applications, used to train several deep learning algorithms with Greek-BERT obtaining the best accuracy scores.

Keywords

Greek patents, Automated Classification, Natural Language Processing, Deep learning, Greek-BERT.

Περίληψη

Τα διπλώματα ευρεσιτεχνίας είναι έγγραφα που περιέχουν τελευταίας τεχνολογίας τεχνικές και επιστημονικές πληροφορίες σχεδόν σε κάθε τομέα της επιστήμης. Οι αιτήσεις διπλωμάτων ευρεσιτεχνίας που υποβάλλονται καθημερινά στον Οργανισμό Βιομηχανικής Ιδιοκτησίας Ελλάδος πρέπει να ταξινομούνται πνευματικά από ειδικούς του τομέα βάσει ιεραρχικής ταξινόμησης. Ως αποτέλεσμα, αυτή η διαδικασία ταξινόμησης είναι εντατική και συντριπτική για τους ειδικούς. Αυτό το έγγραφο προτείνει τη χρήση της Επεξεργασίας Φυσικής Γλώσσας (ΕΦΓ) για την αυτοματοποιημένη ταξινόμηση των διπλωμάτων ευρεσιτεχνίας. Συγκρίνουμε τις πιο σύγχρονες μεθόδους ΕΦΓ που βασίζονται σε Βαθιά Μάθηση για την αυτοματοποιημένη σημασιολογική κατηγοριοποίηση αυτών των εγγράφων. Για το σκοπό αυτό, δημιουργήσαμε ένα σύνολο δεδομένων που αποτελείται από περίπου 70.000 ελληνικές αιτήσεις διπλωμάτων ευρεσιτεχνίας, που χρησιμοποιείται για την εκπαίδευση αρκετών αλγορίθμων βαθιάς μάθησης με το Greek-BERT να λαμβάνει τις καλύτερες βαθμολογίες ακρίβειας.

Λέξεις – κλειδιά

Ελληνικές Πατέντες, Αυτόματη ταξινόμηση, Επεξεργασία Φυσικής Γλώσσας, Βαθιά Μάθηση, Greek-BERT, Νευρωνικά δίκτυα, Τεχνητή νοημοσύνη.

Table of Contents

List of Tables

List of figures

Acronym Index

AI: Artificial Intelligence

ASR: Automated Speech Recognition

BERT: Bidirectional Encoder Representations from Transformers

BOW: Bag of Words

CBOW: Continuous Bag of Words

CNN: Convolutional Neural Network

CPC: Cooperative Patent Classification

DCNN: Deep Convolutional Neural Networks

EMT: Εθνικό Μητρώο Τίτλων

EPD: English Patents Dataset

EPO: European Patent Office

FC: Fully Connected

GAN: Generative adversarial networks

GPD: Greek Patents Dataset

GPU: Graphics processing unit

GPVD: Greek Patent Validation Dataset

IP: Intellectual Property

IPC: International Patent Classification

KNN: K-Nearest Neighbors

LDA: Linear Discriminant Analysis

LSTM: Long-Short Term Memory

MTGP: Machine Translated Greek Patents

NB: Nive Bayes

NER: Name Entity Recognition

NLP: Natural Language Processing

NN: Neural Network

POS: Part of Speech

QA: Quality Assurance

ReLU: Rectified Linear Unit

SVM: Support Vector Machine

USPTO: United States Patent and Trademark Office

WIPO: World Intellectual Property Organization

INTRODUCTION

Patents are important documents in the development and dissemination of technology.

For a patent organization to grant a patent, it must meet three basic criteria: a) novelty, b) inventive step, c) industrial applicability. This process is called prior art search which is a part of the examination phase. Each technological field has its own department within the organization; it consists of experts in each field to evaluate the abovementioned criteria.

Patents, include a huge amount of technical and scientific information. Each patent has bibliographic data which contain the number of the patent, its classification, the inventor, date of filling and date of grant etc. Moreover, each patent has its unique title, abstract and drawings. Lastly, the most important are the claims. All these information is available, and is open access through numerous databases (EPO, USPTO, etc.)

The classification process is used by all patent organizations for the purpose of developing a classified database internationally. Furthermore, classification constitutes a powerful tool that examiners, patent attorneys, researchers and many others use to find similar technical patents. In fact, inventors themselves can, through classification, find the state of the art in their field of interest in order to get ideas or find solutions to their technical problems.

In particular, patent classification taxonomy is consisted of several levels. It starts with section which is the 1st level and class which is the 2nd level. The 3rd level is the subclass, 4th the group and 5th the subgroup (Fig. 1). The IPC eighth edition consists of eight Sections, 129 classes, 639 subclasses, 7,314 main groups, and 61,397 sub-groups [37]. Thus, patent classification is a multi-classification problem with a total of 68,899 classes. Each class/subgroup contains several patents related to the subject of the subgroup. Although there are other types of classification like CPC, a lot of patents are not arranged using these taxonomies and sometimes there not classified from official sources (i.e., national patent offices), so the official and only classification is the IPC.

The subject of this thesis

In recent years, the rapid growth of patents has increased the need for automated processes and data processing systems. The reason to implement such an idea is because Patent offices and third-party organizations which have a large volume of patents, needs such a tool. Classification process takes time and effort as long as resources. Automated classification can save time since it is being done within seconds while an examiner needs 1-3 hours. Moreover, inside an organization, such a tool can be useful as a rooting agent that can redirect the patent in the respective technological field department.

Aim and objectives

The aim of this thesis is to evaluate and compare the contribution of several artificial intelligence methods in automated patent applications classification. This classification system has been adopted and implemented by all national patent offices worldwide. The classification system is the International Patent Classification system (IPC) [37] and World Intellectual Property Organization (WIPO) oversees it. In particular, the main contributions of this thesis are the following:

- To the best of our knowledge, this is the first work on automated Greek patent classification.
- To build a dataset comprised by around 70,000 Greek patents.
- To develop and evaluate several deep learning-based NLP models for patent classification.

Methodology

In the methodology that was used in this research was mostly experimental quantitative analysis. First, a problem formulation was introduced. This states the "what" is this research investigates. After the problem formulation, literature review was conducted, and some approaches were distinguished. Finally, results were introduced and an evaluation on how reliable these results are was done.

Innovation

Considering that Greece has a limited patent culture, not such approach or tools has been developed in the past. Greek technical language and terminology is an unexplored field with many applications and potentials. Thus, this work is considered as innovative, and it opens a new field to be explored.

Structure

Firstly, there will be a brief introduction and description of the patents system. What is a patent, what are the main parts of a patent, in order to give an idea to the reader of patents in general. After having an idea of the patent system, a small introduction to machine learning and deep learning will be presented. The purpose of this chapter is to present the basic ideas and state of art techniques in natural language processing. By understanding these algorithms and approaches, the final problem formulation is introduced. Related work will be presented with similar approaches that was reviewed and which of these approaches were considered to be used in this research. Methodologies are being presented and the approaches that has been followed to solve this problem. Therefore, by implementing all these methodologies, the results are presented with a small discussion in each methodology. Finally, the conclusion summarizes the results of this thesis and future steps are proposed.

1 Patent System

This chapter is a brief description of the patent system and what are the main parts of a patent.

1.1 Patents

A patent is a title of protection with a duration of 20 years granted to the holder for new inventions, which involve inventive activity and are amenable to industrial engineering application. These inventions can be either products, either product production method, or industrial application. Patents are legal documents that gives the legal right to the inventor to use, produce and sell a particular invention; they are being granted by IP authorities all over the world. [38]

For a patent to be granted several criteria must be met:

a) Novelty is considered an invention that does not belong to state of the art.
   The definition of novelty and prior art is absolute, i.e., state of art contains everything known anywhere in the world (worldwide) by written or oral description or by any means or by any person, before the date of submission of the application for grant of the patent (or the priority date claimed by the applicant).
b) Inventive activity is considered to include an invention if, according to judgment of an expert (person of the profession), no it follows in an obvious way from the level of technique.
c) Industrial application is considered to have one invention if its object can be produced or used in any means of productive activity.

1.2 Structure of Patents

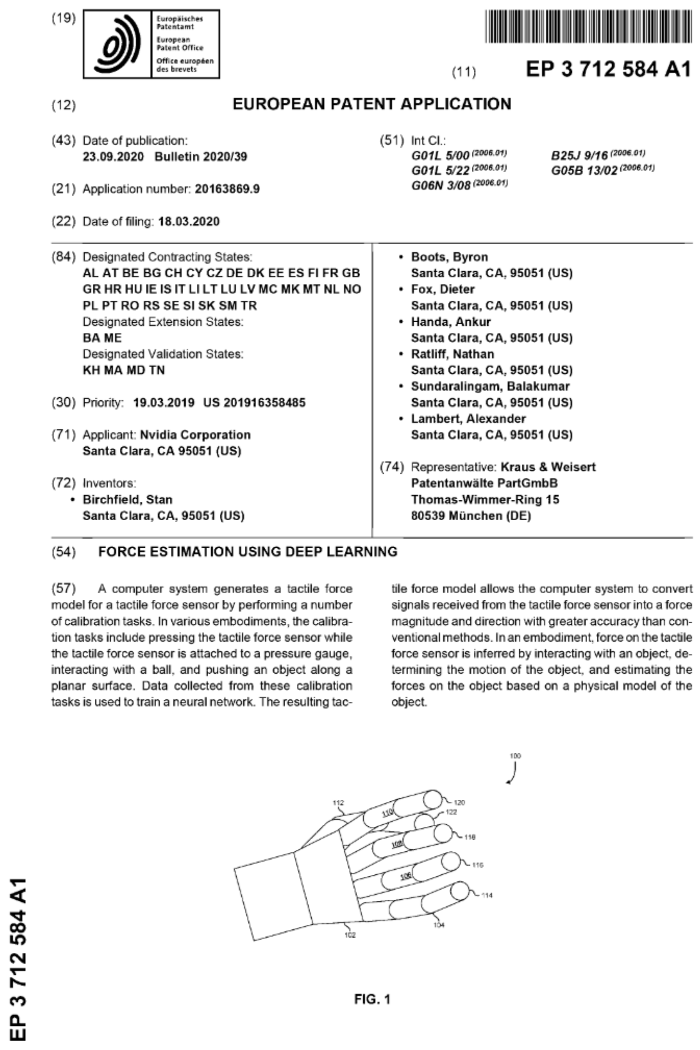
As mentioned before, patents are legal documents that protects inventions. Inventions have technical characteristics and sometimes are very complex. For someone to get a better overview of what is the subject of a patent and what it is protecting, patents are divided in three categories. Description, Claims and Drawings. Along with the publication of a patent, Bibliographic data are also included. In most online databases, there are more information available, like Legal events, Search Report, Patent Family, and other documents regarding the procedure of the application.

The Patent Cooperation Treaty (PCT) is an international agreement that provides a standardized process for filing a single patent application to protect an invention in multiple countries. The rules governing the PCT are contained in the PCT Administrative Instructions and the Common Regulations. These rules outline the procedures for filing, processing, and maintaining PCT applications, as well as the requirements for entering the national phase in each designated country. [39]

1.2.1 Bibliographic Data

Bibliographic data are general information of the patent. In bibliographic data Internationally agreed Numbers for the Identification of Data (INID) is being used as being in the Standard ST.9 [38]. Bibliographic data in patents refer to information about a patent document that is used to identify and locate the patent. These data typically include the name of the inventor or inventors, the title of the invention, the patent number, the filing date, and the publication date.

They may also include other information such as the name of the assignee (the company or individual to whom the patent has been assigned) and the classification of the patent. Bibliographic data are important because they allow researchers and other interested parties to easily locate and access patent documents. It is typically included in patent databases and other resources that are used to search for and retrieve patent information. An example can be seen in Figure 1.1 of a published patent.



(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 3 712 584 A1**

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
23.09.2020 Bulletin 2020/39

(51) Int Cl.:
G01L 5/00 *(2006.01)* ***B25J 9/16*** *(2006.01)*
G01L 5/22 *(2006.01)* ***G05B 13/02*** *(2006.01)*
G06N 3/08 *(2006.01)*

(21) Application number: **20163869.9**

(22) Date of filing: **18.03.2020**

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA ME
Designated Validation States:
KH MA MD TN

(30) Priority: **19.03.2019 US 201916358485**

(71) Applicant: **Nvidia Corporation**
Santa Clara, CA 95051 (US)

(72) Inventors:
- **Birchfield, Stan**
  Santa Clara, CA, 95051 (US)
- **Boots, Byron**
  Santa Clara, CA, 95051 (US)
- **Fox, Dieter**
  Santa Clara, CA, 95051 (US)
- **Handa, Ankur**
  Santa Clara, CA, 95051 (US)
- **Ratliff, Nathan**
  Santa Clara, CA, 95051 (US)
- **Sundaralingam, Balakumar**
  Santa Clara, CA, 95051 (US)
- **Lambert, Alexander**
  Santa Clara, CA, 95051 (US)

(74) Representative: **Kraus & Weisert**
Patentanwälte PartGmbB
Thomas-Wimmer-Ring 15
80539 München (DE)

(54) **FORCE ESTIMATION USING DEEP LEARNING**

(57) A computer system generates a tactile force model for a tactile force sensor by performing a number of calibration tasks. In various embodiments, the calibration tasks include pressing the tactile force sensor while the tactile force sensor is attached to a pressure gauge, interacting with a ball, and pushing an object along a planar surface. Data collected from these calibration tasks is used to train a neural network. The resulting tactile force model allows the computer system to convert signals received from the tactile force sensor into a force magnitude and direction with greater accuracy than conventional methods. In an embodiment, force on the tactile force sensor is inferred by interacting with an object, determining the motion of the object, and estimating the forces on the object based on a physical model of the object.

FIG. 1

EP 3 712 584 A1

Figure 1.1: Example of bibliographic data taken from patent EP3712584A1 from Nvidia Corp regarding an invention for force estimation using deep learning.

1.2.2 Drawings

Drawings in patents are graphical representations of the invention that are included in the patent document. They are used to illustrate how the invention works and to provide additional details about its construction and operation. Drawings in patents may include diagrams, schematics, cross-sectional views, perspective views, and other types of graphical representations. They are typically accompanied by a written description of the invention, and together, the drawings and the written description provide a complete and detailed disclosure of the invention. In continuation of the bibliographic data, an example of drawings of the same patent can be seen in Figure 1.2. The drawings in a patent are an important part of the patent document because they provide a visual representation of the invention and can help to clarify and explain the written description.

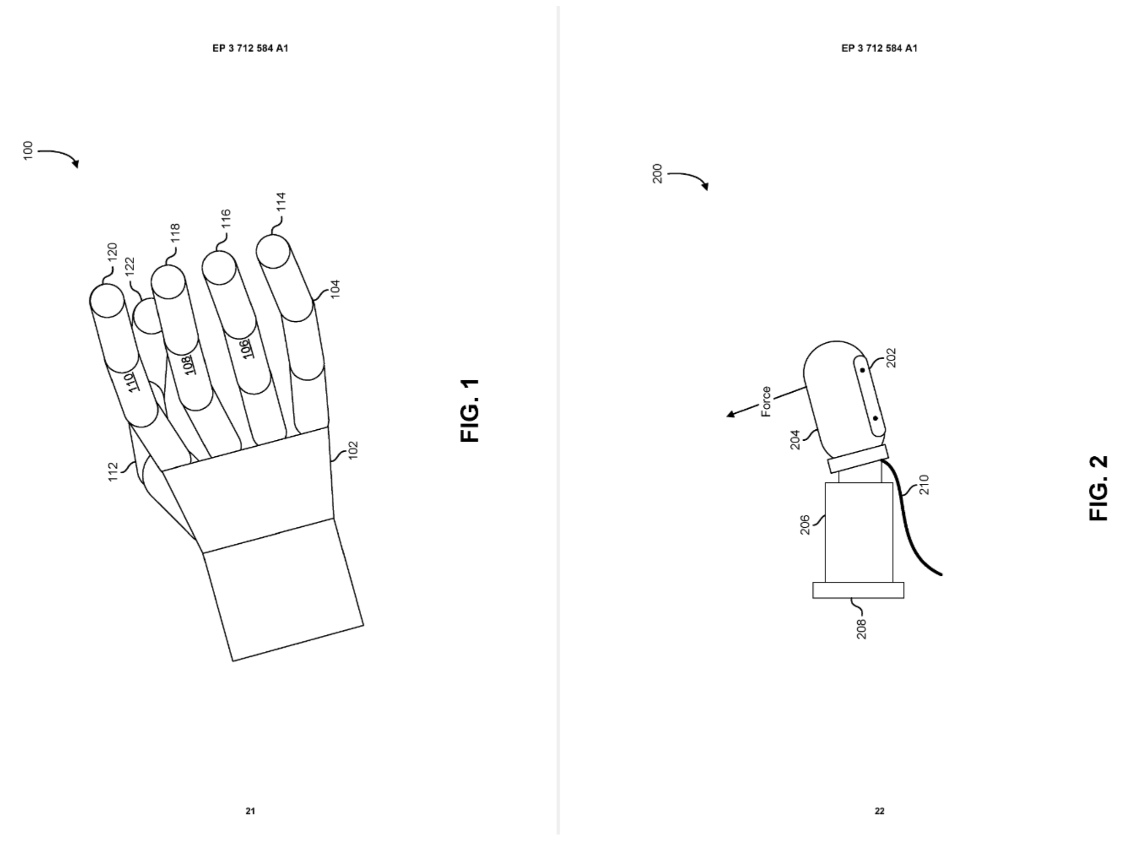


Figure 1.2: Example of drawings taken from patent EP3712584A1 from Nvidia Corp regarding an invention for force estimation using deep learning.

1.2.3 Description

The description in a patent is the written portion of the patent application that describes the invention in detail. It typically includes a written description of the invention, as well as one or more drawings or diagrams illustrating the invention.

The description serves several important functions. It must provide sufficient information to enable someone skilled in the relevant field to understand and practice the invention. It must also clearly and distinctly describe the claimed invention and distinguish it from prior art. Additionally, the description must enable the public to understand the nature and operation of the invention, as well as the manner in which it can be used.

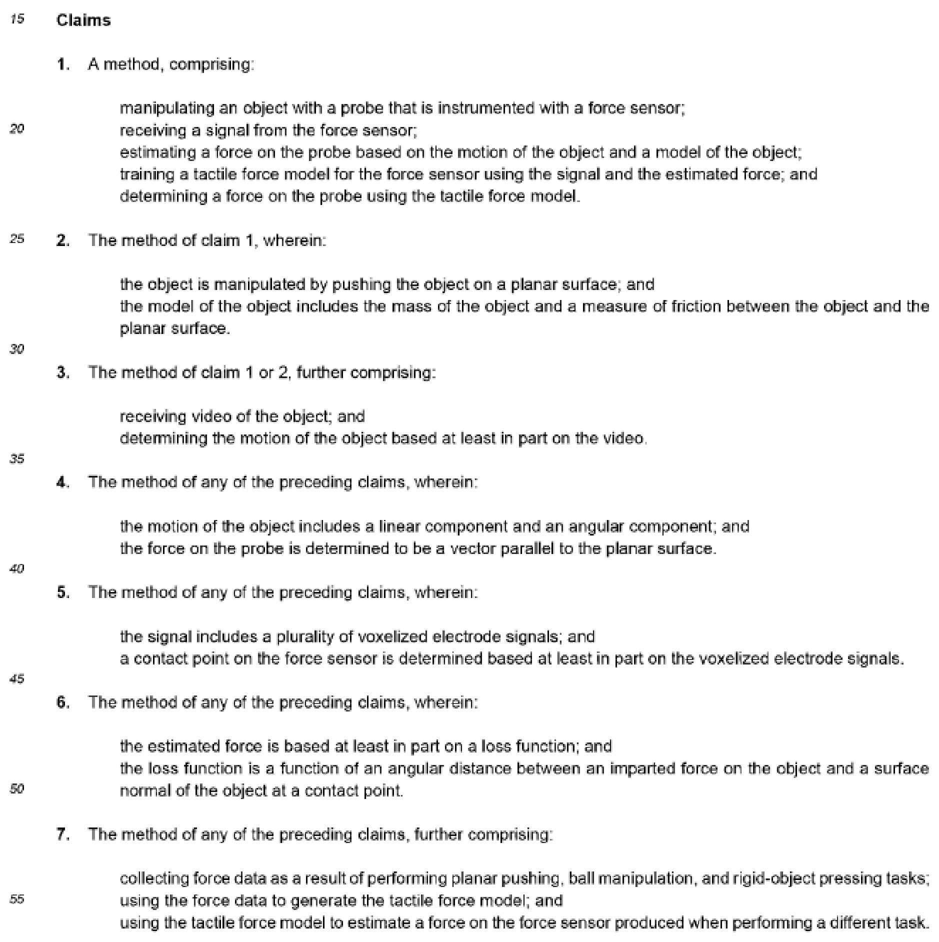
The description is an important part of the patent application process, and it is essential that it be carefully drafted in order to effectively and accurately describe the invention. It is also important to note that the description is a public document, and once the patent is granted, it becomes part of the public record and is available for anyone to read and use.

1.2.4 Claims

The claims in a patent are the legal definitions of the invention that are being protected by the patent. They are the part of the patent that defines the scope of the protection that is granted to the inventor. The claims typically appear at the end of the patent application and are written in a specific format that is designed to clearly and concisely set forth the boundaries of the invention.

There are two main types of claims in a patent: independent claims and dependent claims. Independent claims stand on their own and define the invention in broad terms, while dependent claims are based on and build upon the language of one or more independent claims.

It is important to carefully draft the claims in a patent application in order to adequately protect the invention and clearly define the legal rights of the inventor. The claims, an example of which is shown in Figure 1.3, must be carefully balanced between being broad enough to provide adequate protection and being narrow enough to avoid infringing on the rights of others.



Claims

1. A method, comprising:

   manipulating an object with a probe that is instrumented with a force sensor;
   receiving a signal from the force sensor;
   estimating a force on the probe based on the motion of the object and a model of the object;
   training a tactile force model for the force sensor using the signal and the estimated force; and
   determining a force on the probe using the tactile force model.

2. The method of claim 1, wherein:

   the object is manipulated by pushing the object on a planar surface; and
   the model of the object includes the mass of the object and a measure of friction between the object and the planar surface.

3. The method of claim 1 or 2, further comprising:

   receiving video of the object; and
   determining the motion of the object based at least in part on the video.

4. The method of any of the preceding claims, wherein:

   the motion of the object includes a linear component and an angular component; and
   the force on the probe is determined to be a vector parallel to the planar surface.

5. The method of any of the preceding claims, wherein:

   the signal includes a plurality of voxelized electrode signals; and
   a contact point on the force sensor is determined based at least in part on the voxelized electrode signals.

6. The method of any of the preceding claims, wherein:

   the estimated force is based at least in part on a loss function; and
   the loss function is a function of an angular distance between an imparted force on the object and a surface normal of the object at a contact point.

7. The method of any of the preceding claims, further comprising:

   collecting force data as a result of performing planar pushing, ball manipulation, and rigid-object pressing tasks;
   using the force data to generate the tactile force model; and
   using the tactile force model to estimate a force on the force sensor produced when performing a different task.

Figure 1.3: Example of claims taken from patent EP3712584A1 from Nvidia Corp regarding an invention for force estimation using deep learning.

1.3 IPC Classification

International Patent Classification is the standard classification used in all patents all over the world. This was established by the Strasbourg Agreement 1971. IPC is a hierarchical system, which classifies each patent to the relevant field which they pertain[1]. The reason classification is used is because patents are being documented and they are being deposited continuously and

[1] https://www.wipo.int/classifications/ipc/en/

in large numbers, so in order to store them and also search and retrieve documents, they must be grouped according to the technical content. Because patents are multinational documents with sources all over the world, the need of the classification extends even more. Because abstracts or full text are not available from all countries and also the use of different language is making more difficult to search or understand document, classification introduces a language-independent tool which determines the field of the patent which is available to anyone.

IPC is consisted of several levels. It starts with section which is the 1st level and class which is the 2nd level. The 3rd level is the subclass, 4th the group and 5th the subgroup (Figure 1.4). The IPC eighth edition consists of eight Sections, 129 classes, 639 subclasses, 7,314 main groups, and 61,397 sub-groups [37]. Each class/subgroup contains several patents related to the subject of the subgroup. Although there are other types of classification like CPC, a lot of patents are not arranged using these taxonomies and sometimes there not classified from official sources (i.e., national patent offices), so the only classification which is used from all IP authorities is the IPC.

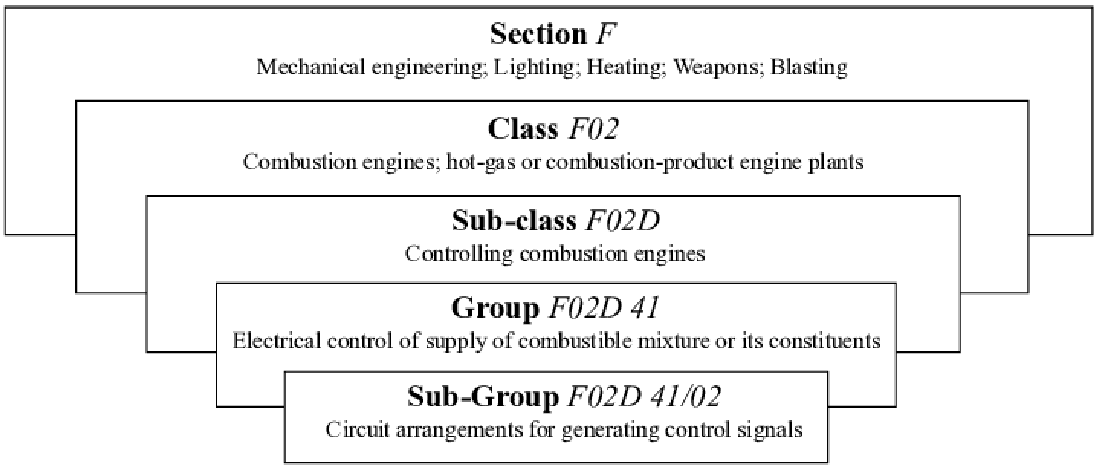
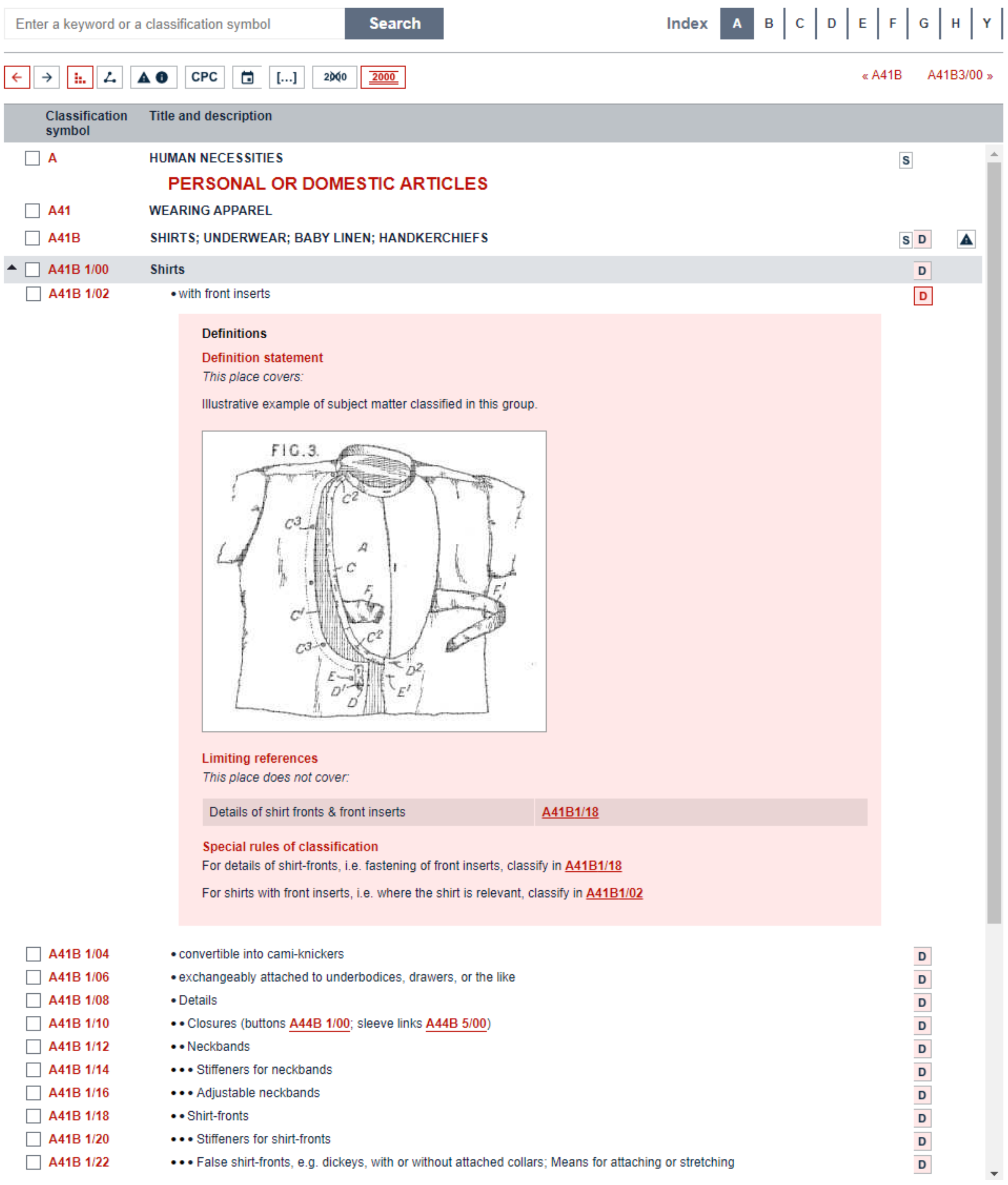

Section *F*
Mechanical engineering; Lighting; Heating; Weapons; Blasting
Class *F02*
Combustion engines; hot-gas or combustion-product engine plants
Sub-class *F02D*
Controlling combustion engines
Group *F02D 41*
Electrical control of supply of combustible mixture or its constituents
Sub-Group *F02D 41/02*
Circuit arrangements for generating control signals

Figure 1.4: IPC structure for each level given a specific example for F02D41/02 in the IPC [26]

There are numerous tools that someone can explore the IPC. One of these, which is also a patent searching tool, is the Espacenet[2]. With this tool, someone can browse the IPC and also search for IPCs entering keywords. Also, in each classification, there are detailed information of what is covers with a representative drawing etc. (Figure 1.4).

[2] https://worldwide.espacenet.com/patent/cpc-browser



Classification symbol	Title and description
A	HUMAN NECESSITIES
	PERSONAL OR DOMESTIC ARTICLES
A41	WEARING APPAREL
A41B	SHIRTS; UNDERWEAR; BABY LINEN; HANDKERCHIEFS
A41B 1/00	Shirts
A41B 1/02	• with front inserts

Definitions

Definition statement

This place covers:

Illustrative example of subject matter classified in this group.

Limiting references

This place does not cover:

Details of shirt fronts & front inserts	A41B1/18

Special rules of classification

For details of shirt-fronts, i.e. fastening of front inserts, classify in A41B1/18

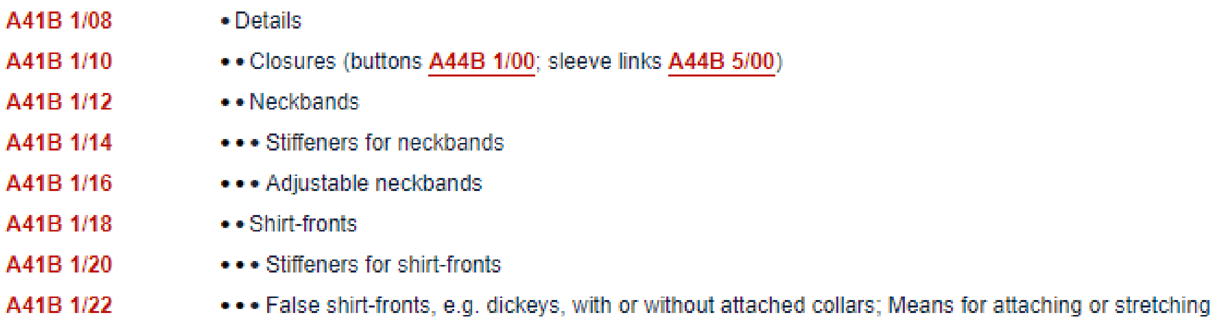
For shirts with front inserts, i.e. where the shirt is relevant, classify in A41B1/02

A41B 1/04	• convertible into cami-knickers
A41B 1/06	• exchangeably attached to underbodices, drawers, or the like
A41B 1/08	• Details
A41B 1/10	• • Closures (buttons A44B 1/00; sleeve links A44B 5/00)
A41B 1/12	• • Neckbands
A41B 1/14	• • • Stiffeners for neckbands
A41B 1/16	• • • Adjustable neckbands
A41B 1/18	• • Shirt-fronts
A41B 1/20	• • • Stiffeners for shirt-fronts
A41B 1/22	• • • False shirt-fronts, e.g. dickeys, with or without attached collars; Means for attaching or stretching

Figure 1.5: Example of description of each IPC[3].

The hierarchy in the IPC is declared with dots. For example, a single dot is higher that two dots and two dots higher than three dots and so on. With this hierarchy, it is known in which fields is containing subfields similarly to set theory in mathematics. More particularly, as shown in Figure 1.6, A41B1/08 contains all the below mentioned classes and A41B1/10 does not contain any other subgroups but A41B1/12 also contains A41B1/16 which is a three-dot subgroup under the two dot A41B1/12.

[3] https://worldwide.espacenet.com/patent/cpc-browser



A41B 1/08	• Details
A41B 1/10	• • Closures (buttons A44B 1/00; sleeve links A44B 5/00)
A41B 1/12	• • Neckbands
A41B 1/14	• • • Stiffeners for neckbands
A41B 1/16	• • • Adjustable neckbands
A41B 1/18	• • Shirt-fronts
A41B 1/20	• • • Stiffeners for shirt-fronts
A41B 1/22	• • • False shirt-fronts, e.g. dickeys, with or without attached collars; Means for attaching or stretching

Figure 1.6: Example of description of each IPC and the dot hierarchy. [Source]

1.4 IP Authorities

Intellectual property (IP) authorities are organizations or agencies that are responsible for administering and enforcing laws related to intellectual property. These laws protect the rights of creators and owners of various types of IP, including patents, trademarks, copyrights, and trade secrets.

There are many different IP authorities around the world, and the specific laws and regulations that they enforce can vary depending on the country or region in which they operate. In the United States, for example, the US Patent and Trademark Office (USPTO) is the federal agency responsible for administering and enforcing IP laws. Other countries have their own IP authorities, such as the European Patent Office (EPO) in Europe, the Japan Patent Office (JPO) in Japan, and the China National Intellectual Property Administration (CNIPA) in China.

IP authorities play a critical role in protecting the rights of IP owners and helping to ensure that IP is used and shared fairly. They also work to promote innovation and creativity by providing a legal framework for the creation and protection of new ideas and creative works.

1.4.1 Greek Registry

The Greek Registry also known as Εθνικό Μητρώο Τίτλων (EMT), allows interested parties to retrieve relevant information on industrial property titles, filed in Greece and published in the Special Bulletin of Industrial Property (EDBI) which is published every month. Also, all Greek patents are published in all open-source databases and are accessible publicly by anyone.

2 Artificial Intelligence and Deep Learning

In this chapter, basic ideas and approaches about machine learning and deep learning in Natural Language processing (NLP) are described. Purpose of this chapter is to give an overview of state of art tools that can be used in NLP.

2.1 Introduction to Artificial Intelligence

To touch the subject of deep learning and NLP, it is first really important to go through the history, techniques and applications of the scientific fields that these sub-fields belong to. These fields are Artificial intelligence and Machine Learning.

Artificial intelligence, or AI in short, encompasses the capacity of machines to execute cognitive functions such as perception, learning, thinking and problem-solving. The benchmark for AI is set to the level of human cognitive abilities in reasoning, communication, and visual processing. AI is now utilized in a vast array of industries, providing a technological advantage for businesses that adopt it. Additionally, AI has the potential to add significant value to industries such as banking, retail and logistics and transportation, by comparison to traditional analytics methods. [8]

One possible use of AI is that for example a company can provide a conversation intelligence service, where every phone call made by a sales professional is recorded, transcribed, and analyzed by the computer, utilizing AI data and suggestions to help create a successful plan. That means that if a company uses AI for its marketing personnel, they can automate repetitive and tedious tasks, allowing the sales representative to focus on developing relationships and nurturing leads. [8]

In summary, AI offers advanced technology for handling complex data that a person alone cannot handle and can lead to increased efficiency and profitability for companies. [8]

2.1.1 History

The concept of AI is not a recent development, and the first significant event in AI history was a summer project organized in 1956 by leaders in various scientific fields. This project was led by John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon, all from reputable institutions such as Dartmouth College, Harvard University and Bell Telephone Laboratories. [31]

There was a significant boom in AI development between 1957 and 1974, as computers performed in greater speed, were cheaper and more households could thus have access to them, and machine learning algorithms improved. ELIZA by Joseph Weizenbaum and General Problem Solver by Newell and Simon were two noticeable early attempts at spoken language interpretation and problem-solving. These projects showed promise and attracted the attention of government entities like the Defense Advanced Research Projects Agency (DARPA), who sponsored AI research at various institutions. Computers and AI had potentially the ability to analyze vast amounts of data, transcribe, and translate spoken language. Of course, such automated skill and work was of great interest to authorities. [31]

In the early days of AI, expectations to such systems and algorithms were really high and a lot of people were optimistic about their capabilities. However, despite some early successes, significant challenges remained in achieving goals such as natural language processing, abstract thinking, and self-awareness. A concrete representation of the timeline is given by the Figure 2.1. For example, in 1970, AI pioneer Marvin Minsky stated in an interview that a machine with the general intelligence of a human could be developed within 3-8 years, but this has yet to be achieved. [31]

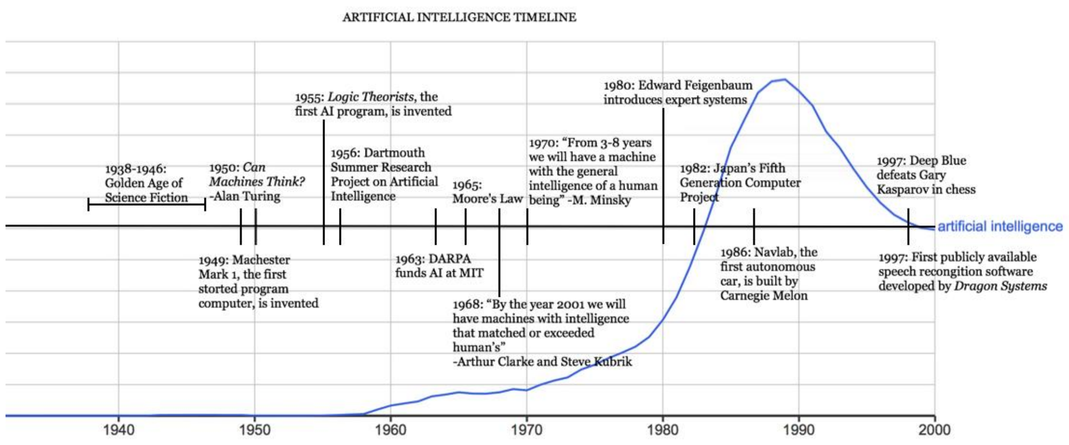


Figure 2.1: Timeline of AI. [31]

The early days of AI faced several challenges, the major one being the limited processing power and storage capacity of computers, which made it difficult to achieve significant progress. For example, successful communication requires the ability to understand the meanings of multiple words and their various combinations. As a result, funding and research slowed down for several years. Even academics had their fair thoughts of doubts regarding AI. For example, Hans Moravec, a doctoral student under John McCarthy, noted that at the time, computers were "still millions of times too frail to exhibit intelligence." [31]

Despite these obstacles, AI continued to advance without much government support or media attention. In the 1990s and 2000s, many of the original goals of AI were achieved. For example, IBM's Deep Blue defeated world chess champion Gary Kasparov in 1997, and speech recognition software from Dragon Systems was integrated into Windows the same year. These developments were major steps forward in the creation of artificially intelligent decision-making software and spoken language interpretation. Additionally, robots like Kismet, which was capable of recognizing and displaying emotions, demonstrated that even human emotions were within the realm of possibility for machines. [31]

2.1.2 Artificial Intelligence Today

It is important to question what has changed and if humanity is becoming any better and more efficient at developing artificial intelligence. One of the limitations in the past was the limited storage capacity of computers, but that is no longer an issue as Moore's Law, which states that computer memory and performance will double every year, has met and in many cases exceeded our needs. This is how Deep Blue was able to defeat Gary Kasparov in 1997 and how Google's Alpha Go recently defeated Chinese Go champion Ke Jie. A pattern of our days that can explain the fluctuations in the progress of AI research is that scientists now push AI's capabilities to the limits of current computational power (computer storage and processing speed) and then wait for the advancements in technology to catch up, as per Moore's Law. [31]

The "big data" era has also allowed researchers to collect large amounts of data that would be difficult for a person to analyze. This has already led to significant success in the use of artificial intelligence in various sectors and aspects of our life, such as technology in general, but also in work-related activities like banking and marketing, but of course also in people's entertainment. Even if the same algorithms and computational methods are used, large amounts of data and powerful computers can allow artificial intelligence to learn through repetition and brute force. Technology has come a long way since the 80's and 90's and a clear result of that is that Moore's Law may be slowing down. Nevertheless, data growth has not. Every day, millions of bytes that represent new information pass right before our eyes, the data that is available for AI models to train on is endless. Thus, advancements in computer science, mathematics, or neurology may overcome the limitations of Moore's Law. [31]

One of the most important subfields of AI today is Natural Language Processing (NLP), the main focus of this essay. NLP focuses on providing computers with the ability to understand written and spoken words in a way similar to humans. [1]

Natural Language Processing (NLP) is a subset of AI that focuses on providing computers with the ability to understand human language in a way that is similar to how humans process it. This field of AI combines several techniques such as statistical models, machine learning, and deep learning together with computational linguistics, which is a rule-based approach to understanding human language. These technologies allow computers to process human language in various forms such as text or audio, and understand the meaning, context, and intent behind the words. This has led to the development of various NLP applications such as language translation, voice commands, and text summarization, which can understand the sentiment and emotions behind the words. This technology is being used in various industries to improve efficiency and automation of business operations, customer service, and other applications. [1]

Some key application areas of NLP are the following:

- Searching
- Machine translation
- Summarization
- Named-entity recognition
- Parts-of-speech tagging
- Information retrieval
- Information grouping
- Sentiment analysis
- Answering queries
- Automated speech recognition (ASR)

2.2 Machine Learning

Machine learning is the field that enables computers to learn based on patterns and prior experience. As a field of computer science, it has existed since 1950, with the publication of A. Turing [1]. Depending on the methods used, machine learning can be divided into supervised learning and unsupervised learning, and there are versions that combine both. In supervised learning, the user trains the program to predict based on known and named classes of data. In contrast, in unsupervised learning, the algorithm creates the classes from unordered, unknown data, forming groups based on how much the data share characteristics with each other. They are mainly used to discover patterns in data sets. AI Chatbots and AI Virtual Assistants use either one or a balanced combination of Supervised Learning and Unsupervised Learning. [1]

2.2.1 Supervised Learning

Artificial intelligence chatbots and virtual assistants are trained on properly labeled data using supervised learning (or tagged). Supervised Natural Language Processing (NLP) models are trained using a process called supervised learning, in which a system is presented with a set of known input data and corresponding desired output. During the training process, the system identifies the optimal mapping function between the input data and the desired output. This mapping function, also known as a model, is then used to analyze new and unseen input data, and accurately predict the corresponding output. This approach allows the system to learn from the past experiences and generalize it to new input data, making the predictions more accurate. This is the basic concept behind supervised learning in NLP, which is widely used in various applications such as language translation, text classification, and sentiment analysis. The use of this approach allows for the system to learn from the past experiences and generalize it to new input data, making the predictions more accurate and efficient. Before Supervised Learning models reach an expected and commonly recognised level of performance, the input-output mapping typically needs to be changed over the course of several iterative optimization cycles. Because of how it learns from training data and how closely it resembles the way of learning while someone (usually a teacher) supervises you and checks your progress and mistakes, this type of learning has retained the label "supervised." [2]

It is important to state that supervised learning models often require a lot of time and technical skills from data scientists and programmers to be scientifically accurately and efficiently designed, scaled, deployed, and of course maintained. For example, supervised learning is a popular approach used in building AI chatbots and virtual assistants. One of the key tasks that these models perform is classifying user input into pre-defined categories known as intents. This task is commonly achieved by utilizing different algorithms such as Support Vector Machine, Random Forest, and Classification Trees. For example, a virtual assistant that helps users find information about flights, might have an intent class called "flight_booking" and the model would be trained to recognize a user's input such as "I want to book a flight" as belonging to that intent class. The use of these algorithms allows the model to learn from a pre-defined set of input-output examples and then generalize it to new unseen inputs, thus enabling the chatbot or virtual assistant to respond to users with the right intent. [2]

This approach is widely used in various industries such as customer service, e-commerce, and healthcare to improve the user experience and automate various tasks. As all these classes are pre-defined, it can be easily concluded how much time and attention is needed from the programmer’s part to keep the model running as intended, which may be in some cases a timewasting activity. The accuracy attained by those techniques is of course quite impressive, but one drawback is that they can only cover the intent classes for which there is labeled data available for training. [2]

2.2.2 Unsupervised Learning

Unsupervised Learning, which allows for effective learning using unlabeled data (no labeled data is required for training), has become increasingly popular in academia and industry as a way to overcome the limitations of Supervised Learning (no need for data scientists or high-technical expertise). Large amounts of unlabeled text in digital form are widely available, but labeled datasets are often costly to build or acquire, particularly for standard NLP tasks like PoS tagging or Syntactic Parsing. This is a significant advantage over supervised learning. Unsupervised learning models have the automation and intelligence needed to operate independently and automatically identify information, structure, and patterns in the data, making Unsupervised NLP excel.

Clustering is a method of unsupervised learning that is commonly used in AI chatbots and virtual assistants to group together semantically similar user inputs, in order to identify underlying shared user intents. Unlike supervised learning, where the model is trained on pre-defined input-output examples, clustering does not require predefined categories and instead groups similar inputs together to form new classes of intents. This approach is useful in quickly identifying and validating new user intents without the need for manual annotation. [2]

Clustering (such as K-mean, Mean-Shift, Density-based, Spectral clustering, etc.) and association rules approaches are the most commonly used forms of unsupervised learning in advanced AI chatbots and virtual assistants. Clustering is typically used to automatically group semantically related user utterances together in order to quickly identify and validate a common user intent (deriving a new class, not classifying into an existing class). Association rules mining, which seeks to deduce correlations between characteristics directly from data, also uses unsupervised learning. This method is often used to automatically extract pre-existing dependencies between named entities from user input, dependencies of intents across a set of user input that is part of the same user/system session, or dependencies of questions and answers from conversational logs that record user interactions with live agents during the problem-solving process.

Although unsupervised learning has many benefits and is highly automated, it typically performs less accurately and reliably than supervised learning. Thus, one main goal of AI models today, especially chatbots and virtual assistants, is to achieve the balance between supervised and unsupervised learning, meaning that it aims to be as accurate as the models using the first method, and on the same time as self-automated as the models using the latter method. [2]

2.2.3 Semi-supervised Learning

Semi-supervised learning refers to a learning problem (and algorithms created for the learning problem) where a model must learn and make predictions on fresh instances from a small part of annotated examples and a large number of not annotated examples. As a result, it is a learning challenge that falls in the middle of supervised and unsupervised learning.

In cases where tagging instances is difficult or expensive, semi-supervised learning algorithms are necessary. If a semi-supervised learning algorithm can perform better than a supervised learning algorithm fitted solely to annotated training instances, it is successful. Algorithms for semi-supervised learning can typically meet this low standard.

Finally, semi-supervised learning can be used to compare inductive and Transductive learning. A learning algorithm that obtains information from labeled training data and applies that knowledge to fresh data, such as a test dataset, is known as inductive learning. Transductive learning is the process of learning from labeled training data and generalizing to easily accessible unlabeled (training) data. Both types of learning tasks can be handled by a semi-supervised learning system. [14]

2.2.4 Reinforcement Learning

The basis of the reinforcement learning machine learning training technique is rewarding desired actions and/or penalizing undesirable ones. The goal of an agent created through reinforcement learning is to learn from its mistakes and act based on its each time unique environment. [4]

In this method, developers provide a way to reward desired actions and penalize undesirable behaviors to motivate the agent. The agent is trained to seek maximum overall reward in the long run to achieve the best possible outcome. This approach also prevents the agent from getting stuck on small tasks and will learn to avoid negative and look for positive. This learning strategy has been adopted by Artificial intelligence (AI) as a technique to control unsupervised machine learning through rewards and penalties.

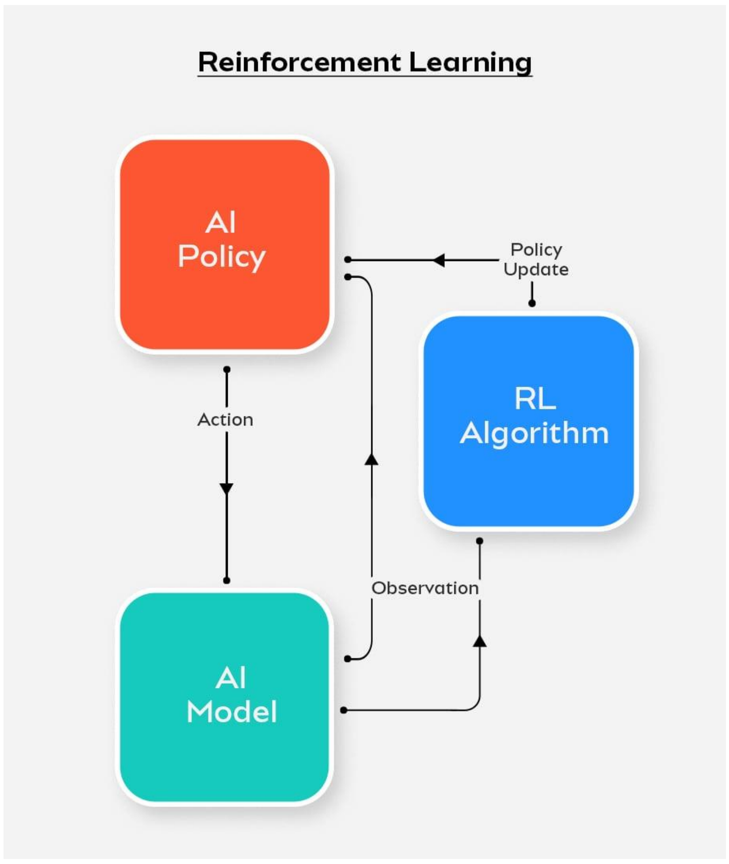


Figure 2.2 Basic function of a Reinforcement Learning method [4]

The AI model can learn the best behavior using reinforcement learning to maximize the likelihood of a successful outcome. The AI model learns what should ideally be the most profitable behavior patterns through feedback from its environment. A closed loop visual representation can be seen in Figure 2.2.

Several studies show how deep reinforcement learning algorithms can be used to solve practical NLP issues. Following are some examples of reinforcement learning's uses in NLP: [4]

- Goal-oriented Dialogue Generation
- Machine Translation
- Abstractive Text Summarization
- Question Answering

2.3 Deep Learning

Deep learning is a subset of machine learning that uses multi-layered neural networks to try and mimic the way the human brain functions, but falls short of matching it, allowing it to "learn" from large amounts of data. Additional hidden layers can improve accuracy, even if a neural network with just one layer can still make approximate predictions.

Many Artificial Intelligence (AI) applications and services are powered by deep learning, which enhances automation by performing mental and physical tasks without human intervention. Deep learning powers both established and emerging technologies, such as voice-activated TV remote controls, digital assistants, credit card fraud detection, and self-driving cars.

Long ago, the bulk of approaches to research NLP issues relied on labor-intensive, hand-crafted features and shallow machine learning models. The limited encoding of linguistic information in traditional models resulted in issues like the curse of dimensionality (high-dimensional features). However, recent advancements in neural-based models, specifically the use of word embeddings (low-dimensional, distributed representations) have led to their success and popularity in outperforming conventional machine learning models such as SVM or logistic regression in various language-related tasks. [13]

2.3.1 How Deep Learning started?

When Walter Pitts and Warren McCulloch developed a computer model based on the neural networks of the human brain in 1943, deep learning had its beginnings.

They imitated the mental process using a set of mathematical formulas and algorithms they named "threshold logic." Since then, there have only been two important interruptions in the development of deep learning. Both have anything to do with the infamous AI winters.

In 1960, Henry J. Kelley is credited with creating the fundamentals of a continuous back propagation model. Stuart Dreyfus created a simplified variant in 1962 that just used the chain rule. Although the idea of back propagation—the backward propagation of faults for training purposes—was there in the early 1960s, it was cumbersome and ineffective, and it wouldn't be put to good use until 1985.

In 1965, Alexey Grigoryevich Ivakhnenko, who created the Group Method of Data Handling, and Valentin Grigoryevich Lapa, author of Cybernetics and Forecasting Techniques, made the first attempts to create deep learning algorithms. They employed models with polynomial activation functions (complex equations), which were then statistically examined. The top statistically picked features from each layer were then passed on to the following layer (a slow, manual process). [20]

Kunihiko Fukushima employed the first "convolutional neural networks." Fukushima created neural networks with several convolutional and pooling layers. He created the Neocognitron artificial neural network in 1979, which had a hierarchical, multilayered design. The computer may "learn" to recognize visual patterns thanks to this design. Although taught using a reinforcement technique of repeated activation in numerous layers, the networks matched contemporary models and grew stronger with time. Furthermore, Fukushima's design allowed for human adjustment of significant elements by raising the "weight" of specific connections.

A variety of neural networks have been realized with the use of top-down connections and innovative learning techniques. The Selective Attention Model can distinguish between and recognize distinct patterns when more than one pattern is displayed at once by switching its focus from one to the other. (The method that many of us employ when multitasking.) A contemporary Neocognitron can not only recognize patterns that are incomplete (for instance, a missing fifth) but can also finish the image by inserting the necessary data. One could call this "inference."

At Bell Labs in 1989, Yann LeCun gave the first actual demonstration of backpropagation. On read "handwritten" digits, he applied back propagation and convolutional neural networks. In the end, handwritten checks with numbers on them were read using this method.

Around this period, the second AI winter (1985–1990) began, which had an impact on deep learning and neural network development. A number of excessively optimistic people overstated the "immediate" potential of artificial intelligence, disappointing investors in the process. The outrage was so great that the term "artificial intelligence" became debunked. Thankfully, some people kept working on AI and DL, and tremendous progress was made. The support vector machine was created by Vladimir Vapnik and Dana Cortes in 1995. (a system for mapping and recognizing similar data). LSTM (long short-term memory) for recurrent neural networks was developed in 1997, by Sepp Hochreiter and Juergen Schmidhuber.

When computers began processing data more quickly and graphics processing units (GPUs) were created in 1999, deep learning took its next big evolutionary stride. Over a ten-year period, faster processing, with GPUs processing images, boosted computational speeds by 1000 times. [20]

The potential and difficulties of data growth were described as three-dimensional in a research report published in 2001 by META Group, now known as Gartner. According to the report, the variety of data sources and types has expanded as a result of the growth in data volume and speed. This was a warning to get ready for the impending invasion of big data.

In 2009, Stanford AI scientist Fei-Fei Li created a free collection of over 14 million tagged photos called ImageNet, as labeled images were necessary to "train" neural networks but were not easily accessible. Professor Li stated that the use of large data would transform machine learning, as learning is driven by data.

Thanks to a significant increase in GPU speed, convolutional neural networks could be trained without the layer-by-layer pre-training in 2011. As processing speed continued to increase, it became clear that deep learning had significant advantages in terms of efficiency and speed. One example of this is AlexNet, a convolutional neural network that won multiple international contests for its architecture in 2011 and 2012, using rectified linear units to increase dropout and speed.

2014 saw the introduction of the Generative Adversarial Neural Network (GAN). Ian Goodfellow developed the GAN program. Two neural networks compete with one another using GAN. In order to win, one network must successfully duplicate a photograph and deceive the other into thinking it is real. [20]

2.3.2 NLP Representation

Before neural approaches emerged and addressed some of the issues with conventional machine learning models, such as the curse of dimensionality, hand-crafted features were predominantly employed to model natural language tasks.

The so-called distributional hypothesis states that words with similar contexts have comparable meanings. This is the foundation for distributional vectors, also known as word embeddings. Word embeddings are pre-trained using a shallow neural network on a task where the goal is to predict a word based on its context. A neural language model is demonstrated in the following Figure 2.3:

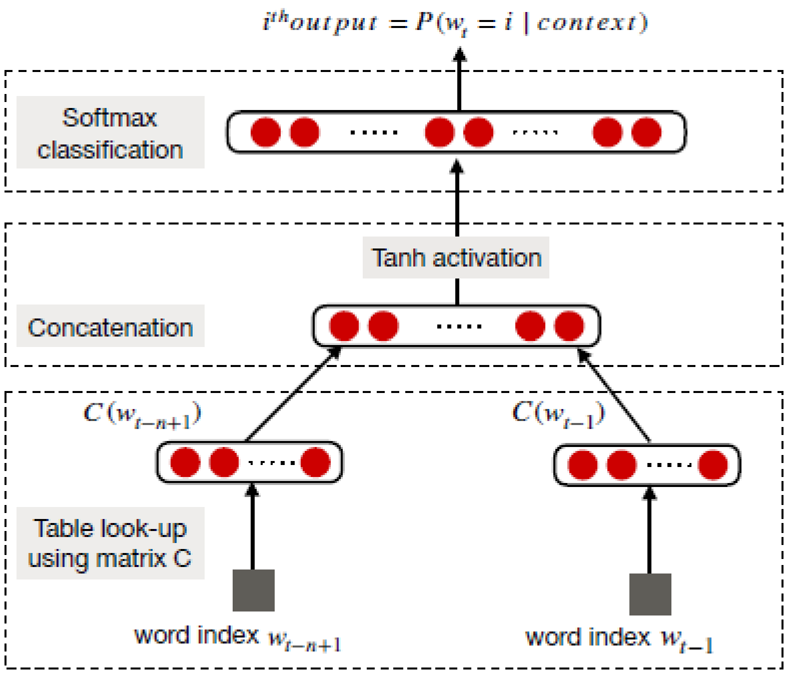


Figure 2.3 Simple Neural Language Model demonstration. [10]

In a range of NLP tasks, including sentiment analysis and sentence compositionality, the word vectors have a tendency to embed syntactical and semantic information and at the moment they consist state of the art tools.

Distributed representations were extensively employed in the past to investigate a variety of NLP problems, but it wasn't until the continuous bag-of-words (CBOW) and skip-gram models entered the field that they began to acquire traction. They gained popularity because to their rapid creation of high-quality word embeddings and their application in semantic compositionality (e.g., "man" + "royal" = "king"). [10]

The CBOW and skip-gram models were proposed by Mikolov et al. around 2013 as part of the Word2Vec technique, which is an algorithm that uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence. The technique represents each distinct word with a unique set of numbers called a vector, which are chosen to capture the semantic and syntactic qualities of words. This allows a simple mathematical function, cosine similarity, to indicate the level of semantic similarity between the words represented by those vectors.

A Word2vec model can be trained using two methods: hierarchical softmax and negative sampling. The hierarchical softmax method uses a Huffman tree to reduce calculation and approximate the conditional log-likelihood the model aims to maximize. The negative sampling method, on the other hand, approaches the maximization problem by minimizing the log-likelihood of sampled negative instances. According to the authors, hierarchical softmax works better for infrequent words, while negative sampling works better for frequent words and with low dimensional vectors. Additionally, as the training epochs increase, hierarchical softmax becomes less effective.

The goal of CBOW, a neural technique to building word embeddings, is to calculate the conditional probability of a target word given the context words in a certain window size. The aim of Skip-gram, on the other hand, is to anticipate the surrounding context words (i.e., conditional probability) given a central target word. Skip-gram is a neural technique to creating word embeddings. The word embedding dimension for both models is calculated (unsupervised) by calculating prediction accuracy.

It can be difficult to get vector representations for words like "hot potato" or "Boston Globe" when using word embedding techniques. Since these phrases don't capture the mixture of meaning of the individual words, we can't directly combine the vector representations of the individual words. When considering larger phrases and sentences, it becomes even more challenging.

The use of smaller window sizes results in comparable embeddings for opposing terms like "good" and "bad," which is not desired, especially for applications where this separation is crucial, like sentiment analysis. This is another problem with word2vec models. Word embeddings have another drawback in that they depend on the application in which they are employed. It has been investigated to retrain task-specific embeddings for each new task, although this is typically computationally expensive and can be dealt with more effectively utilizing negative sampling. Word2vec models also have other issues, like the failure to account for polysemy and other biases that could appear in the training data.

In order to do tasks like named-entity recognition (NER) and parts-of-speech (POS) tagging, it is helpful to examine morphological information in words, such as letter combinations. For morphologically complex languages like Portuguese, Spanish, and Chinese, this is also helpful. These embeddings help with the unknown word problem since we are no longer expressing sequences with vast word vocabularies that need to be condensed for effective computation purposes because we are studying text at the character level.

The long-term effects of both character-level and word-level embeddings have been questioned, despite the fact that they have been successfully used in a variety of NLP applications. As an illustration, Lucy and Gauthier recently discovered that word vectors fall short in their ability to adequately represent the various components of conceptual meaning that underlie words. In other words, it is asserted that it is impossible to comprehend concepts behind words alone by distributional semantics. [10]

2.3.3 Neural Networks

2.3.3.1 Dense Neural Networks

In a deep neural network, Dense Connections, also known as Fully Connected Connections, are a type of layer that utilize a linear operation where every input is connected to every output through a weight. This design leads to a large number of parameters, specifically n_inputs x n_outputs (1.1), making it computationally expensive for larger networks. Furthermore, this type of connection can also lead to overfitting, where the model becomes too complex for the given data, due to the large number of parameters. Despite these challenges, Dense Connections are widely used in deep learning models due to their ability to capture complex patterns in the data. These characteristics of the DNN will be analyzed in the following paragraphs. [9]

The architecture of the deep learning model can be thought of as its layers. Different kinds of layers may be utilized in the models. Based on their features, each of these several strata has a particular significance. Convolutional layers are used in image processing, LSTM layers are frequently used in time series analysis and NLP difficulties, etc. The last stages of the neural network use a layer known as a dense layer, also known as a fully linked layer. This layer aids in altering the output's dimensionality from the layer before so that the model may more easily establish the relationship between the values of the data it is working with. [9]

Although a densely connected neural network normally won't perform the best on text data, it's an excellent place to start to understand how to create and test deep learning models for text. This kind of network architecture can be utilized as a link between bag-of-words methods used in simpler classification and regression models and methods other than bag-of-words, which are mainly employed in LSTM and CNN models. Word sequences and placements can be taken into account in addition to word counts thanks to deep learning. [9]

Figure 2.4 shows a feed-forward neural network topology with many connections. The first hidden layer is tightly (in this example, totally) connected to the input when it enters the network all at once. A layer is "hidden" in the sense that the input and output layers handle its connection to the outside world. Only the following layer is connected to the neurons in any given layer. Variable hyperparameters of the model chosen by the practitioner include the number of layers and nodes inside each layer. [9]

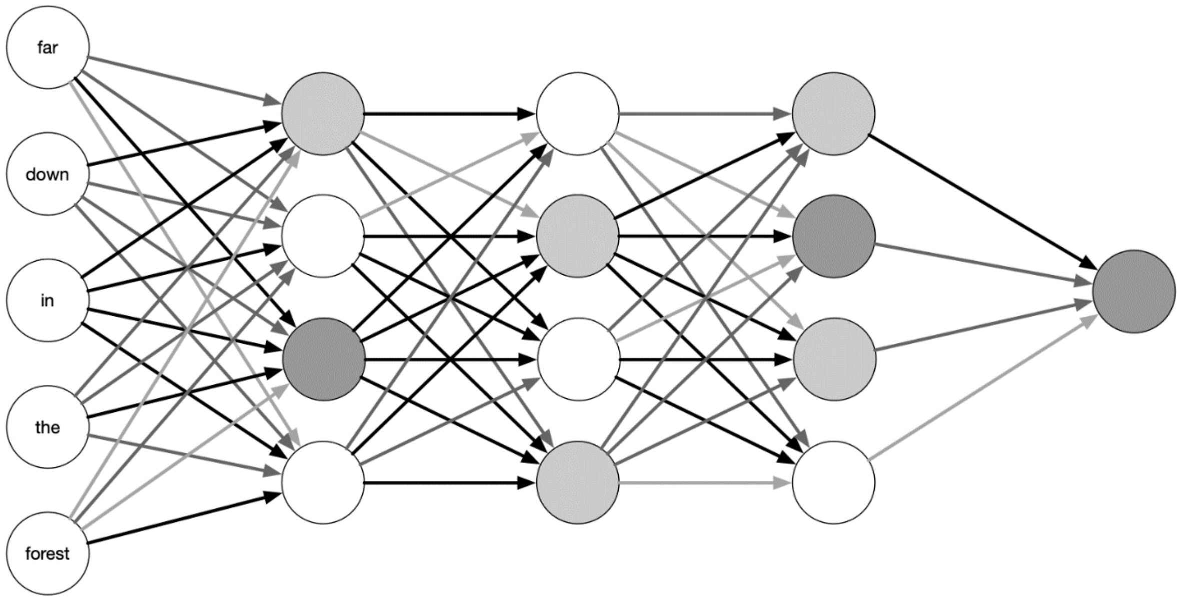


Figure 2.4 High-level diagram of a feed-forward neural network [43]

The input units are depicted with words in Figure 8.1, however this is not a completely accurate representation of a neural network. As neural networks only function with numeric variables, these words will actually be represented by embedding vectors.

A densely connected layer in a neural network refers to a layer where every neuron is connected to every neuron in the preceding layer. This type of layer is commonly used in artificial neural networks. During the forward pass, each neuron in the preceding layer sends signals to the neurons in the dense layer, which perform matrix-vector multiplications. The output of the previous layers is multiplied by the dense layer's weights, and the result is passed to the next layer. This is only possible when the number of columns in the output of the previous layer is equal to the number of rows in the dense layer's weight matrix. [43]

The general formula for a matrix-vector product is shown in figure 1.5:

$$A\mathbf{x} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix}.$$

Figure 2.5: General formula for a matrix-vector product. [43]

A is a (M x N) matrix, and x is a matrix with a diagonal of 1, as in Figure 2.5. Backpropagation is a widely used algorithm in the training of feedforward neural networks. It uses gradient descent to optimize the network's parameters, allowing for the efficient update of the weights and biases of the neurons in the dense layers. This process is done by calculating the gradient of the loss function with respect to the network's weights for a single input or output. The dense layers, which are typically used in artificial neural networks, output an N-dimensional vector, and the dimension of these vectors decreases as it moves through the layers. To adjust the dimension of the vectors, each neuron in a dense layer is used. [43]

Every neuron in the thick layer receives information from every neuron in the preceding layers. Therefore, it can be claimed that if the layer above produces a (M x N) matrix by aggregating the results from each neuron, this output flows through the dense layer, where a dense layer should have N neurons.

Dense neural networks are the simplest network architecture that can be applied to fit classification models for text data, and they serve as a useful transitional model to understanding the more complicated model designs that are more frequently used in practice for text modeling. These models require different preprocessing than more straightforward models since they have more parameters than those simpler models. Tokenization enables the development of modeling features that accurately reflect the original text's tokenization order. By doing this, a model may be able to learn from patterns in sequences and order, which is not achievable with more straightforward models.

In DNN, some fine control, like manually creating features using domain expertise, is forgone in favor of feature engineering in the anticipation that the network will eventually learn critical features on its own. The researcher still has some control over feature engineering because they still choose how to normalize and tokenize data before the tokens are sent into the network. [9]

2.3.3.2 Convolutional Neural Networks

The most popular deep learning architectures for image processing and image recognition are convolutional neural networks (CNNs). Given their dominance in the realm of vision, it makes sense that efforts to implement them in many machine learning domains would be made.

A CNN is essentially a neural-based method that represents a feature function that is used to extract higher-level information from constituent words or n-grams. The resulting abstract features have proven useful for a variety of tasks, including sentiment analysis, machine translation, and question-answering. One of the first scholars to use CNN-based frameworks for NLP problems was Collobert and Weston. Their approach's objective was to use a look-up table to convert words into vector representations, which produced a crude word embedding method that learned weights during network training (see Figure 2.6 below). [10]

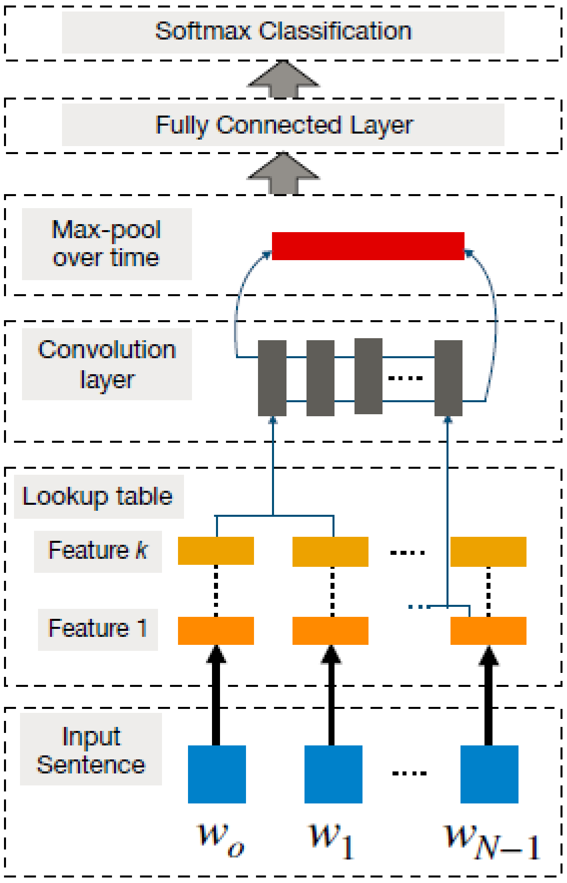


Figure 2.6: Basic CNN architecture. [10]

Sentences are first tokenized into words in order to do sentence modeling with a basic CNN. These words are then changed into a word embedding matrix (also known as an input embedding layer) of d dimensions. The next step is to add convolutional filters to this input embedding layer, which entails applying a filter with every window size that can be used to create a feature map. Afterwards, a max-pooling operation is applied, which performs a max operation on each filter to produce an output of a defined length and lower the output's dimensionality. And the final sentence representation is created by that process.

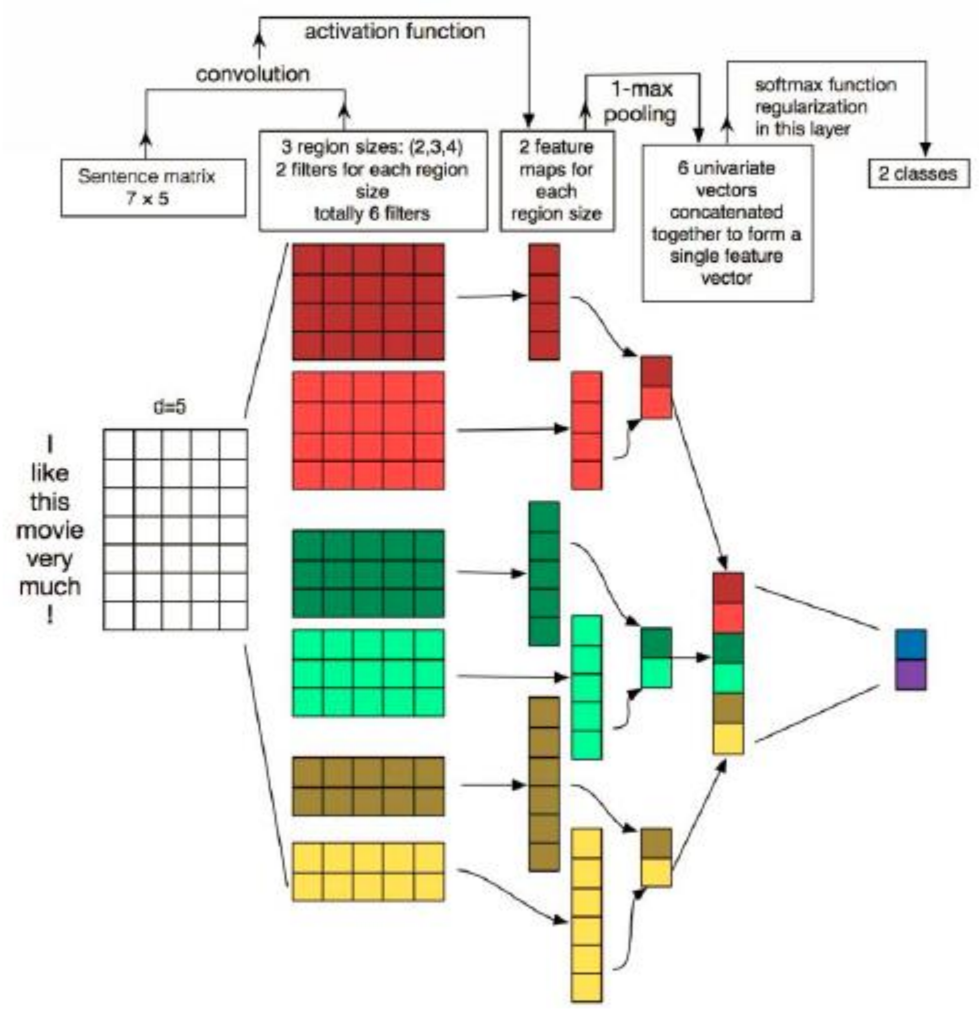

activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 x 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I
like
this
movie
very
much
!

Figure 2.7: Example of Convolutional Neural Networks in NLP and description of its algorithm. [32]

The way that each word (token) is represented as three-dimensional word vectors is the first thing that stands out. An example is shown in Figure 2.7 of how a sentence can be introduced into a CNN network. Then, using one step (also known as a stride), a weight matrix of 3x3 is slid horizontally over the sentence, catching three words at a time. A filter is made up of this weight matrix and an activation function, similar to those found in feed-forward neural networks. The activation function ReLU (rectified linear unit) is mostly utilized in CNNs and deep neural nets because of specific mathematical features.

Returning to image classification, it is generally believed that each filter can identify a variety of features in an image, and that the deeper the filter, the more likely it is to identify complex details. For instance, the first filters in your Convnet will identify basic features like edges and lines, but the features at the very back of the network may be able to identify specific animal species. None of the filters are hardcoded in this process. The weights of these filters will be learned from the data thanks to backpropagation. [32]

The next important step is to calculate the output (convolved feature). The convolution is typically performed by a small kernel, also known as a filter, which is moved across the input data, and a dot product is calculated between the kernel and a small region of the input data at each position. The dot product is then used to create a new feature map, also known as an activation map Figure 2.8 below.

The calculation of a convolution can be broken down into the following steps:

- Initialize a kernel (filter) of a specific size, this is the parameters that will be learned during the training.
- Slide the kernel over the input image, one pixel at a time, starting from the top left corner.
- At each position, perform element-wise multiplication between the kernel and the overlapping region of the input image.
- Sum the results of the element-wise multiplications to get a single scalar value.
- This scalar value is then placed on the corresponding position in the output feature map, also known as the convolved feature.
- Repeat the process for all positions on the input image to create the entire feature map.
- Repeat the above process for all kernels (filters) in the layer to create multiple feature maps.

It's worth noting that the convolution operation can be done in different ways, for example, using padding to handle the edges, or using strides to reduce the computation. Also, the convolution operation can be performed in parallel across multiple filters in a single pass, this is called multiple convolutions. [32]

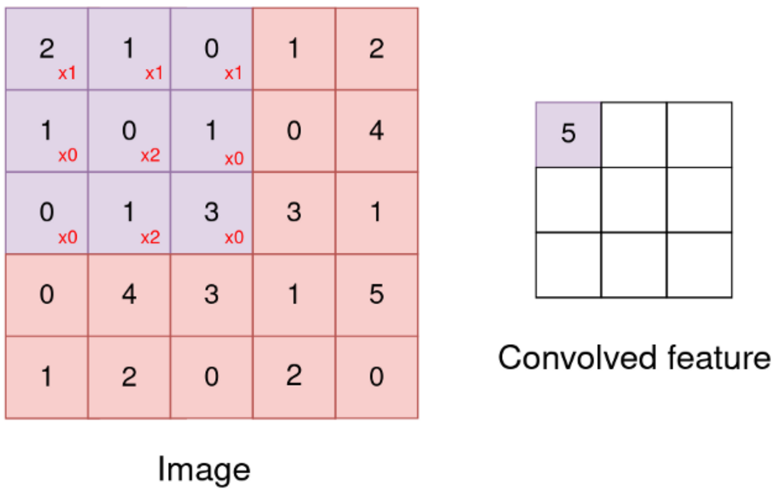


Figure 2.8: Representation of convolved feature in CNNs and how its calculated. [32]

When applied to text a filter that slides by 3 strides horizontally across the window in 1-Dimension will be used:

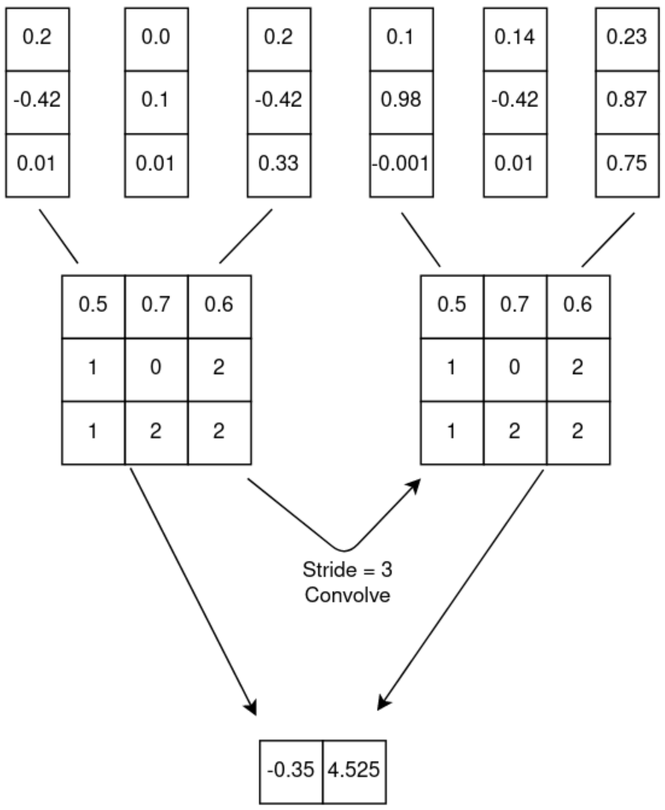


Figure 2.9: Output calculation in CNNs regarding text in convolved procedure. [32]

The output size of the convoluted feature calculation may be less than the input size. It is also not difficult to envision scenarios in which the filter does not perfectly fit the matrix with a specific number of slides. Padding can be used in two different methods to address these issues:

- Add zero vectors to the edges on the outside (zero-padding)
- disregard the matrix row or column that does not fit the filter (valid padding)

In CNNs, pooling is the same as dimension reduction. The main concept is to determine a number that best represents the output by subdividing the output layers. This works so well because it allows the algorithm to learn lower-order representations of the data while using fewer parameters. Common types of pooling are the following:

- Max pooling
- Average pooling
- Sum pooling

The input from the prior pooling and convolutional layers is fed into the final fully connected layer, which subsequently performs a classification task. [32]

Other NLP tasks like NER, aspect identification, and POS can be researched by making the aforementioned basic CNN more complicated and modifying it to make word-based predictions. This necessitates a window-based method, where each word is given consideration within a specified size window of its nearby words (sub-sentence). The training goal of a standalone CNN, also known as word-level classification, is to predict the word in the center of the window after it has been applied to the sub-sentence.

The inability of basic CNNs to simulate long distance dependencies, which is critical for many NLP tasks, is one of their weaknesses. In order to solve this issue, CNNs have been combined with time-delayed neural networks (TDNN), which allow for a wider contextual range to be trained simultaneously. Dynamic convolutional neural networks are additional helpful varieties of CNN that have demonstrated success in a variety of NLP applications, including sentiment prediction and question type classification (DCNN). When performing phrase modeling, a DCNN employs a dynamic k-max pooling technique in which filters may dynamically span variable ranges. [32]

Aspect identification, sentiment analysis, brief text categorization, and sarcasm detection are some of the more sophisticated tasks that CNNs have been utilized for. Other tasks include sentiment analysis and sarcasm detection. However, several of these researches claimed that in order to use CNN-based techniques on microtexts like Twitter tweets, outside information was required. Question-answer representations, query-document matching, speech recognition, machine translation (to some extent), and other tasks have also shown CNN to be effective. On the other hand, for automatic text summarization, a DCNN was utilized to hierarchically learn to collect and synthesize low-level lexical information into high-level semantic notions.

However, CNNs struggle to maintain sequential order and represent long-distance contextual information. CNNs are useful because they can mine semantic hints in contextual windows. Such learning is better suited to recurrent models. [10]

2.3.3.3 Recurrent Neural Networks

RNNs are the best tools for handling issues where the sequence matters more than the specific components.

In essence, an RNNs is a fully connected neural network that has had some of its layers refactored into loops. This loop often involves an iteration over the concatenation or addition of two inputs, a matrix multiplication, and a non-linear function. [7]

The following tasks are among those that RNNs excel at when used with text:

- Sequence labelling
- Natural Language Processing (NLP) text classification
- Natural Language Processing (NLP) text generation

Time series forecasts and other sequence predictions that are not dependent on images or tabular data are other jobs that RNNs are proficient at handling.

RNNs in fact possess an internal memory that enables the prior inputs to influence the predictions made in the future. Knowing the preceding words in a sentence makes it much simpler to predict the subsequent word in that sentence with more accuracy.

The order of the elements is frequently as significant as or more important than the preceding item in jobs best suited to RNNs. [7]

RNNs are specialized neural-based methods for processing sequential data that are efficient. The results of earlier computations are used to conditionally apply a computation to each occurrence of an input sequence by an RNN. A fixed-size vector of tokens that are sequentially (or one at a time) supplied to a recurrent unit serves as a typical representation of these sequences. A straightforward RNN framework is shown in the Figure 2.10 below. [10]

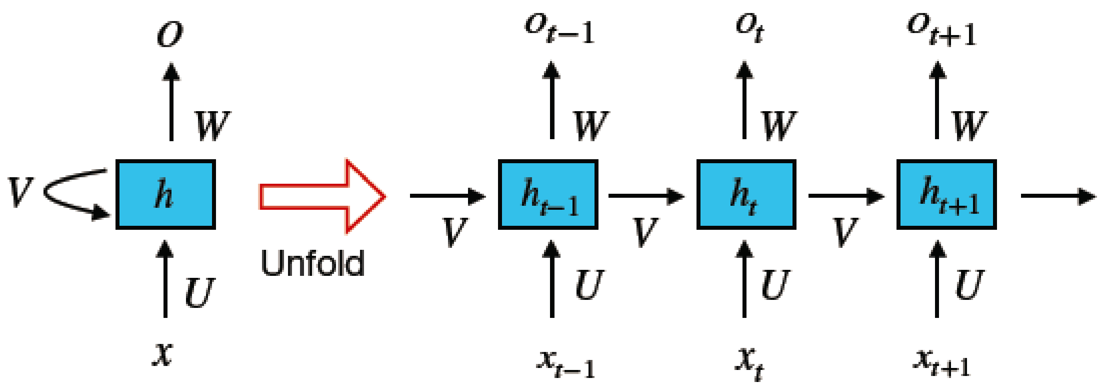


Figure 2.10: Simple recurrent neural networks framework. [10]

The ability of an RNN to remember the outcomes of earlier computations and employ that knowledge in the present computation is its key strength. RNN models can thus be used to represent context dependencies in inputs of any length in order to properly compose the input. RNNs have been used to research a variety of NLP problems, including language modeling, picture captioning, and machine translation, among others.

An RNN model can perform some natural language tasks as well as or even better than a CNN model, although it is not always superior. This is because they represent quite dissimilar features of the data, which limits their usefulness to the semantics necessary for the task at hand. [10]

RNNs normally accept one-hot encodings or word embeddings as input, but occasionally they also accept abstract representations built, for example, by a CNN model. The vanishing gradient problem, which affects simple RNNs, makes it challenging to learn and adjust the parameters in the earlier layers. Later, many variations were developed to get around this restriction, including long short-term memory (LSTM) networks, residual networks (ResNets), and gated-recurrent networks (GRU).

An LSTM consists of three gates (input, forget, and output) and uses a combination of the three to determine the hidden state. Although GRUs only have two gates and are less sophisticated than LSTMs, they are more effective. According to a study, it is challenging to determine which gated RNNs are more efficient, and they are often chosen based on the available computational power. Different LSTM-based models have been put out for sequence-to-sequence mapping (through encoder-decoder frameworks) that are appropriate for applications including text summarization, question answering, simulating human dialogues, machine translation, and more.

LSTMs are designed to address the problem of long-term dependence. Unlike traditional recurrent neural networks, LSTMs are able to maintain a long-term memory, which is their inherent behavior, rather than struggling to learn it.

Recurrent neural networks (RNNs) consist of a series of repeating neural network modules. In a typical RNN, the repeating module is usually composed of a single layer, such as a tanh layer. However, LSTMs have a different structure for their repeating module. Instead of a single layer, LSTMs have four neural network layers that interact in a specific way to form a chain-like topology.[6]

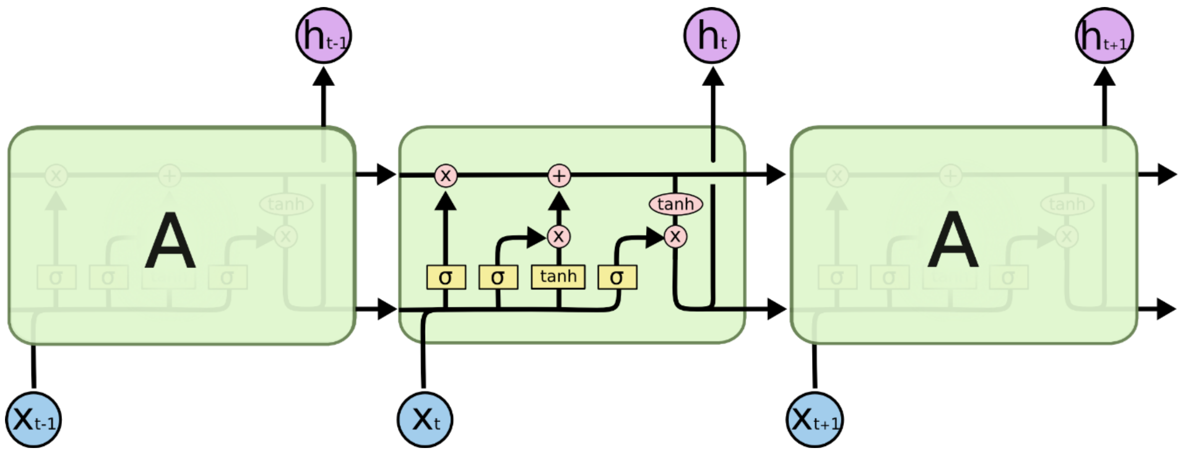


Figure 2.11: The repeating mechanism (cell state) in an LSTM that contains four interacting layers. [6]

The cell state (Figure 2.11) is a crucial variable in LSTMs, which works like a conveyor belt by allowing information to pass through it with minimal modifications. LSTMs can control the flow of information in the cell state through gates, which are composed of a pointwise multiplication operation and a sigmoid layer. The sigmoid layer generates values between 0 and 1, which determine the amount of each component that is allowed to pass through the gates.

Most common applications of RNNs in NLP are [10]:

- Word-level classification (e.g., NER)
- Language modeling
- Sentence-level classification (e.g., sentiment polarity)
- Semantic matching (e.g., match a message to candidate response in dialogue systems)
- Natural language generation (e.g., machine translation, visual QA, and image captioning)

2.3.3.4 Attention Mechanism

In natural language processing (NLP), attention mechanism is a technique used to focus on specific parts of input sequences when processing them. Attention mechanisms allow neural networks to selectively focus on certain parts of the input, rather than using the entire input equally. Attention models are commonly used in tasks such as machine translation, text summarization and question answering, where it's important to understand the relationship between different parts of the input. Attention mechanisms use a set of weights called attention scores, which are learned during training, to determine which parts of the input should be given more importance. These attention scores are calculated by comparing the current input with all other parts of the input, and are then used to weight the input, so that the model pays more attention to the most important parts. [27]

In essence, the attention mechanism is a method that was developed in order to allow the decoder component of the aforementioned RNN-based framework to employ the most recent hidden state in addition to context vector information that was computed using the input hidden state sequence. This is especially useful for activities where some alignment between the input and output text is necessary.

The attention mechanism was first introduced by Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio in their 2014 paper "Neural Machine Translation by Jointly Learning to Align and Translate". The attention mechanism was developed as a way to improve the performance of neural machine translation models. Attention allows the model to focus on specific parts of the input when generating the output, rather than using the entire input equally. This approach proved to be very effective and has since been applied to many other NLP tasks, such as text summarization, question answering, and image captioning. The attention mechanism has been further developed and improved upon by many researchers in the field of NLP and deep learning. [27]

Bahdanau et al.'s attention mechanism can be broken down into the following steps:

- First, the input sequence is passed through an encoder network to generate a set of hidden states. These hidden states represent the meaning of each word in the input sequence.
- Next, the decoder network generates a set of hidden states for the output sequence.
- The attention mechanism then uses the hidden states from the encoder and the decoder to calculate the attention scores. This is done by concatenating the decoder hidden state and each encoder hidden state, and then passing the concatenated vector through a feedforward neural network, which produces the attention scores.
- The attention scores are then used to weight the encoder hidden states. This is done by taking a weighted sum of the encoder hidden states, where the weights are the attention scores.
- The weighted sum is then passed through a tanh activation function to produce the attentional context vector.
- Finally, the attentional context vector is concatenated with the current decoder hidden state, and the concatenated vector is passed through a feedforward neural network to produce the final output.
- The above process is repeated for each time step in the output sequence.

It's worth noting that this is one of the implementations and there are many variations of attention mechanism that has been proposed. Some of them have been proposed and developed after Bahdanau's attention mechanism.

An RNN was used by Bahdanau et al. for both the encoder and decoder. Although the information may not always be related in a sequential manner, the attention process can be re-formulated into a universal form that can be applied to any sequence-to-sequence (abbreviated to seq2seq) activity.

Basically, the generalized attention mechanism works by comparing each key in the database with a query vector that corresponds to a specific word in the input sequence. This is used to determine the relationship between the word being examined and the other words in the sequence. The values are then scaled based on the attention weights, which are calculated from the scores, to focus on the relevant terms in the query. As a result, the word being examined receives an output of attention. [27]

Machine translation, text summarization, image captioning, conversation production, and aspect-based sentiment analysis have all used attention mechanisms successfully. Attention mechanisms come in many different shapes and varieties, and they continue to be a crucial topic of study for NLP researchers looking into a range of applications. [10]

The widely used models for sequence transduction, such as machine translation, are based on complex neural networks that have both encoder and decoder components. These models, such as the Transformer, use an attention mechanism to connect the encoder and decoder, which enables them to outperform previous models by a large margin. The Transformer is a new type of model that does away with recurrence and convolution in favor of attention methods. According to experiments on machine translation tasks, the Transformer models are of higher quality, more parallelizable and faster in terms of training time. On the WMT 2014 English-to-German translation challenge, the Transformer model achieved 28.4 BLEU, outperforming previous models by more than 2 BLEU. It also achieved a new single-model state-of-the-art BLEU score of 41.8 on the WMT 2014 English to French translation problem after training for 3.5 days on eight GPUs. The Transformer is also found to be successful in English constituency parsing with both large and small training data, showing its ability to generalize to different tasks. [3]

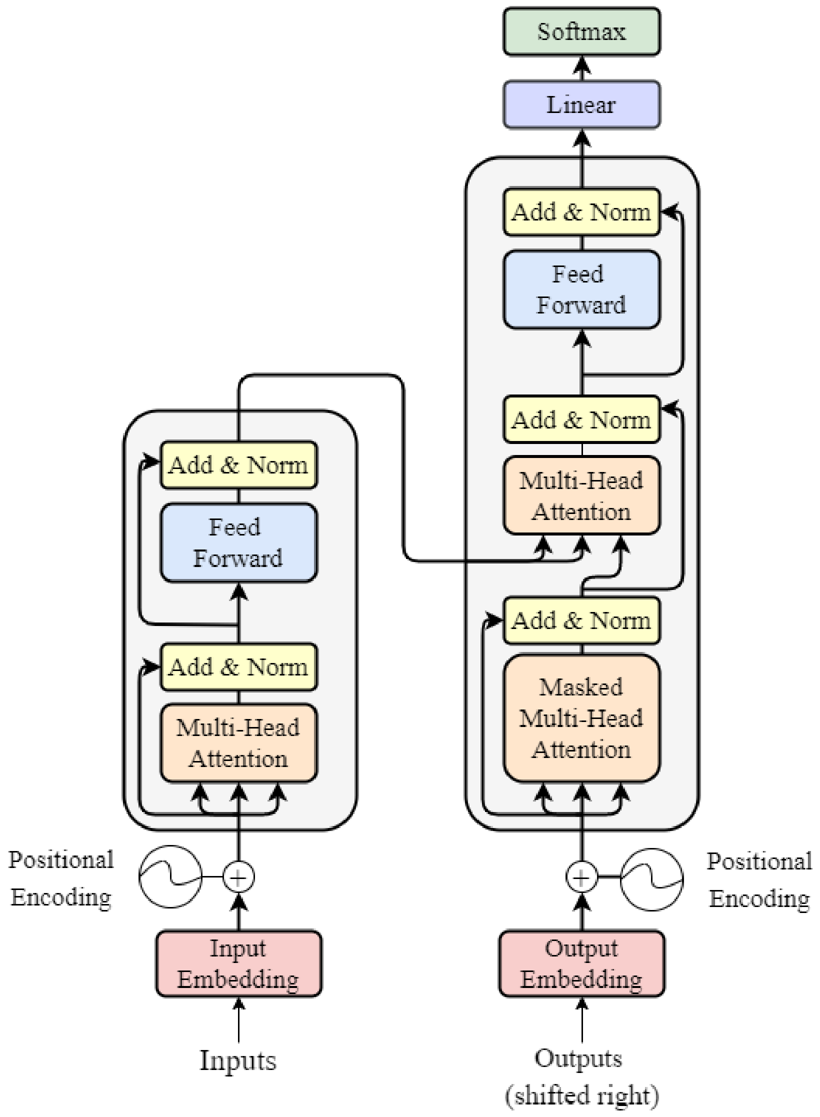


Figure 2.12: The Transformer model architecture. [3]

The Transformer, which architecture can be seen in Figure 2.12, is the first sequence transduction model based solely on attention, substituting the recurrent layers, most frequently found in encoder-decoder designs, for the multi-headed self-attention.

The Transformer may be trained for translation jobs far more quickly than recurrent or convolutional layer-based systems. The transformer attained a new state of the art on the WMT 2014 English-to-German and WMT 2014 English-to-French translation tasks. The best model performs even better than all previously published ensembles in the first task.

In addition to investigating local, restricted attention techniques to effectively handle massive inputs and outputs like graphics, audio, and video, The Transformer will be expanded to problems involving input and output modalities other than text. Another research objective for the Transformer is to reduce generation's sequential nature. [3]

2.4 Related Work on Patent Classification

Many researchers in this field have tried different methodologies in this matter. In this thesis we review the most effective and relevant to the automated classification approach. Since we are talking about text processing, we are referring to Natural Language Processing problems.

2.4.1 Data Sources

Data source is the most "slippery slope" in this classification problem. Due to this research, the findings were that in every single trial for maximizing accuracy and method development different set of data were chosen. This dissimilarity in data makes it hard to compare methods and results that are presented in each approach. There are some well-known datasets, but since technology evolves and patents are being filled every day, there is a constant need in updating these datasets over and over. Such datasets are CLEF-IP[4], USPTO[5] and WIPO-Alpha[6].

2.4.2 Preprocessing

General preparations are taking place from, almost, all researchers. Tools such as tokenization, stopwords removal, word stemming, converting lower case [29] are in every data preparation research, which was evaluated.

2.4.2.1 Bag of Words

Bag of Words (BOW) is a technic which was introduced by Z. Harris in 1954 [11], this was introduced by C.J. Fall and his team in 2003 with a tool called "Rainbow" [5] and it was very helpful in Patent classification approaches since 2013 and in 2017 a team from University of Mato Grosso in Brazil implemented a technic called Continuous Bag of words for a better data preparation in Long Short-Term Memory (LSTM) models [24]. In 2018, S. Li and his team stated that word vectorizing technics would outperform BOW in various ways [23].

2.4.2.2 Embeddings

In NLP embedding aka word vectorization, is a technic widely popular. In this field of patent classification, as no exception, word vectorization is taking place in every single trial for a

[4] http://www.ifs.tuwien.ac.at/~clef-ip/download-central.shtml

[5] https://www.uspto.gov/ip-policy/economic-research/research-datasets

[6] https://www.wipo.int/classifications/ipc/en/ITsupport/Categorization/dataset/wipo-alpha-readme.html

better solution to the problem. In this research, the first who used an embedding technic was the Beijing Information Science and Technology University [40]. This is almost 3 years after "Word2Vec" was published by Tomas Mikolov [35]. Surprisingly, 3 years after a team from Alber-Ludwig University of Freiburg used in their approach 3 systems, "GloVe", "Word2Vec" and "FastText" [21]. Finally, latest different approach was from a team in Seoul University of Science and technology which used a package named "KoRpus" in language R [22]. In June 2021 L. Fang introduced the latest system "Patent2Vec" which is the latest effort in patent embeddings [22].

2.4.3 Methodologies

After reviewing a large number of articles in the patent classification area using Artificial Intelligence and deep learning methods, the findings are quite impressive. Besides that, a small number of researchers are currently on this subject, most recent techniques were used.

In 2003 a team of researchers did an early approach in automated patent classification using Naïve Bayes (NB), K-Nearest Neighbors (KNN), Support Vector Machine algorithms (SVM) and SNoW [25]. They used three different measures of success. Top prediction, Three Guesses and All categories which all of them was a probability in which class a patent would be more fitted in. For comparison reasons we will assess the top prediction category, and, in this area, they were evaluating 3 types of data taken from patents. Titles, Claims and First 300 words. The results were 55% with NB and SVM in 300 words. The rest was below 45%. Lastly in 2011 I. Christopher and hit team implemented a logistic regression which surpassingly outperformed previous methods with a precision of 84% [12] but due to different datasets and computational capabilities those performances can't be compared. I. Christopher and his team also did an improvement in linear SVM algorithm with 92% on 1 level classification and 61% on 2 level classification [12]. The latest effort on SVMs was by J. Yun and Y. Geum in 2020 were they tried an SVM algorithm with radial basis function kernel. Also, they perform Latent Dirichlet Allocation (LDA) method in subgroup level and the results were fantastically good. They manage to achieve 78% precision. So LDA made a huge difference in SVM models [22].

2.4.3.1 Deep Learning

Neural Networks: I. Christopher and his team in 2011 also developed a Neural Network (NN) model. They achieved a Precision of 95% but 59% recall and 71% f1-score which indicates an overfit problem [12]. In 2016, B. Xia and his team in the University of Beijing proposed a new method with Restricted Boltzmann machine which looks promising, but results were not presented [40].

LSTM: After recognition of NB, SVM and KNN weaknesses, M.T. Grawe and his team developed and implemented LSTM method for training. As mentioned before, the combination of Word2Vec and LSTM achieved an accuracy of 63% but 63% recall and 62% f1-score and 66% precision [24]. It is obvious that this is a more stable model. Later in 2019 S. Mustafa developed a system that they named it DeepL4Patent achieved 74% accuracy 92% precision and 63% Recall with 75% f1-score [29].

Convolutional Neural Networks: Last but not least, S. Li in 2018 implemented CNN and word embeddings which they named DeepPatent. They used a quite large dataset, and they achieved 83% Precision and over 80% recall [23].

2.4.4 Hypertuning, Optimization & Transfer Learning

After all these attempts through the years, a team in University of Freiburg in 2019 with L. Abdelgawad in charge attempted a to optimize all these models and compare them.

The results were quite impressive. Two Datasets were chosen, Pat16 and Wipo-alpha [21].

Table 2.1 Accuracy (%) for each model

Method	Pat16 @100	Wipo-Alpha
Linear SVM	68.8	41.0
NB	65.5	33.0
CNN	80.5	49.5
ULMFiT	82.8	49.7
BERT	-	53.4

In conclusion, it is clear from the results in Table 2.1 that transfer learning with ULMFiT and BERT outperform all other methods so far. For Hyperparameter optimization they used a system called BOHB [28]. Also, the best accuracy was achieved using FastText word embeddings.

Finally, J. Lee and J. Hsiang, in 2020 they did a tuning on BERT specifically on patent classification and their results are what we can call state of the art. They used besides IPC a more complex system called CPC (Cooperative Patent Classification) which is similar to IPC, but they manage to achieve 84.26% of precision and 66.8% F1-score. May not be the best in total numbers but the dataset they used was over 11 million patents.

3 Experimental Set-Up for Greek Patent Classification

3.1 Problem formulation

In recent years, the rapid growth of patents has increased the need for automated processes and data processing systems. The reason to implement such an idea is because Patent offices and third-party organizations which have a large volume of patents, needs such a tool. Classification process takes time and effort as long as resources. Automated classification can save time since it is being done within seconds while an examiner needs 1-3 hours. Moreover, inside an organization, such a tool can be useful as a rooting agent that can redirect the patent in the respective technological field department.

Therefore, the final problem formulation and the research topic of this Thesis is "*Development of a classification and routing agent for Greek patents using state of art deep learning tools and methods*".

3.2 Dataset of Patents

3.2.1 Greek Patent Dataset

Greek patents are documents hard to be processed due to the fact that a large number among them have been drawn up by non-certified "patent agents". In order to solve this problem, we needed to find very specific processes to extract the most useful information. Our database source is the Greek National Patent Registry[7] (Εθνικό Μητρώο Τίτλων) EMT where we extracted published patents from the year 2000 until today and we also used some published patents from the EPO that were human translated in Greek. Our dataset consists of 69,020 patents. Although patents have a lot of data and information available, in EMT only the title, abstract and IPC/CPC are in digital form, other sections like description and claims are in digitization process. The only available and useful data that could be extracted were the abstract, title and IPC (used as a label).

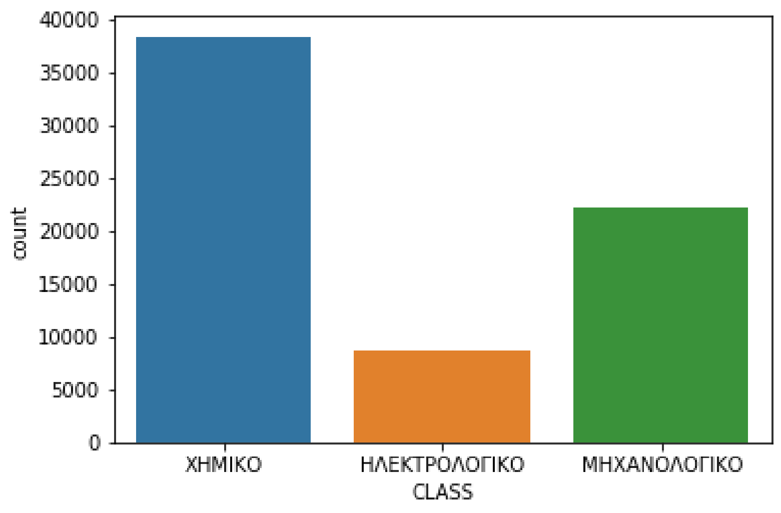


Figure 3.1: Graph of counting 3 classes of the Greek dataset.

[7] Αναζήτηση Εθνικού Μητρώου Τίτλων (obi.gr) Greek Patent Registry

Since patents are technical documents, we divided our patent applications into three major classes based on their scientific field. Mechanical, electrical and chemical/bio-technological are three major categories that have their own semi-unique vocabulary (GPD-3). All patentable inventions can be divided into these three categories. Figure 3.1 or more precisely the

Table 3.1, presents the number of patents for each class. To be able to effectively include each patent document in one of these three classes, we considered the IPC system down to subgroup level, and we reclassified accordingly each document into the abovementioned three classes.

Table 3.1: Number of GPD-3 in each class.

Class	Patents
Mechanical	*22,076*
Electrical	*8,642*
Chemical	*38,302*

A second approach was implemented by separating patent data into 8 classes (GPD-8) (Table 3.2). More specifically, from the IPC classification system we used the first level. As abovementioned the IPC is constructed in five hierarchical levels. Section, class, subclass, group and subgroup. Section is consisted by 8 Sections, A, B, C, D, E, F, G and H. We used the 1st letter of the classification, e.g. (A63B3/63) the (A) corresponds to the section level.

Table 3.2: Number of GPD-8 in each class level of the IPC.

Class		Patents
A	*Human Necessities*	*23,525*
B	*Performing Operations; Transporting*	*9,491*
C	*Chemistry; Metallurgy*	*19,386*
D	*Textiles; Paper*	*594*
E	*Fixed Constructions*	*3,242*
F	*Mechanical Engineering; Lighting; Heating; Weapons; Blasting*	*4,370*
G	*Physics*	*3,701*
H	*Electricity*	*4,711*

The reason those two different types of classification were chosen is because the Greek IP Office has three departments (mechanical, electrical, chemical) in accordance with the nature of the invention and the routing tool must be able to find the right department in which the patent needs to be routed. The second 8 class system gives the field according to the IPC classification which helps the examiner understand in which field the patent is referring to. For example, A class is “Human Necessities” which can contain and mechanical related inventions and electrical related inventions.

Class Weights

Class weights are used to assign a larger weight to the under-represented class during training, so that the model pays more attention to it, and tries to reduce the imbalance. By assigning higher weights to the under-represented class, the model is encouraged to make more accurate predictions for it, and thus improve its overall performance.

There are different ways to assign class weights, some common methods include:

- Manual: Assign class weights manually by setting them to a specific value, such as the inverse of the class frequencies in the dataset.
- Automatic: Calculate class weights automatically based on the class frequencies in the dataset, such as the sklearn's compute_class_weight[8] function.

In summary, class weights in training neural networks are used to adjust the importance of different classes during training. This is particularly useful when the dataset is imbalanced and to ensure the model pays more attention to the under-represented class and reduce the imbalance, thus improving the overall performance of the model. All class weights can be seen in Appendix B.

3.2.2 English Data

Another dataset that was used was a similar dataset with English patents (EPD-8). Patents are mostly in English. In order to get something that would be able to compare results in a bigger scale, a dataset with 2,240,748 patents was used. Table 3.3 describes in detail how many patents are in each class. This dataset was taken from google patents and more specifically from Big Query9 that google uses. To extract this dataset, SQL was used through Google Colab[10] since they were a lot of limitations in space and requests since this is a limited tool from Google[11]. Similarly, IPC was used for classification in an 8-class. Since internal data that we used for GPD-3 are not available and the classification for 2,240,748 patents would take a lot of time, only 8-class was used.

Table 3.3: Number of English patents in each class level of the IPC.

Class		Patents
A	*Human Necessities*	*347,688*
B	*Performing Operations; Transporting*	*501,566*
C	*Chemistry; Metallurgy*	*217,877*
D	*Textiles; Paper*	*30,189*
E	*Fixed Constructions*	*110,042*
F	*Mechanical Engineering; Lighting; Heating; Weapons; Blasting*	*216,307*
G	*Physics*	*423,828*
H	*Electricity*	*393,238*

[8] https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html

[9] BigQuery is a serverless, highly scalable, and cost-effective cloud data warehousing solution provided by Google Cloud. https://cloud.google.com/bigquery#section-1

[10] Google Colab is a free cloud-based platform that allows for collaboration, editing, and running of Jupyter notebooks. https://colab.research.google.com/

[11] Google is a multinational technology company specializing in Internet-related services and products that include online advertising technologies, search engines, cloud computing, software, and hardware. https://www.google.com/

3.3 Preprocessing

As previously mentioned, Greek patents have a lot of noisy information, thus, preprocessing is extremely critical in this approach. Many methods were performed to extract the most useful information out of our data. We also cleaned our data from punctuation and diacritics, and we also uncased all the words. Finally, we limited our documents to the first 200 words in each sample and our corpus was the 15,000 most frequent words.

3.3.1 Stopwords

Stopwords are a set of commonly used words in a language that are usually removed from text data before processing because they are unlikely to contain useful information. These words are considered "stop words" because they do not add significant meaning to the text, and they can be considered as "noise" in the data. Examples of English stopwords include "a," "an," "the," "and," "in," "on," etc. The list of stopwords can vary depending on the specific application or the language being used.

For cleaning purposes, NLTK[12] python library was used which has Greek stopwords, but it was insufficient. Then we took from our corpus the top 15,000 words, and we created a new list of stopwords especially for patents and technical documents.

3.3.2 Stemming

Stemming is a process that reduces words to their base form, also known as the root or stem of the word. This is done by removing the suffixes and prefixes from the word, leaving only the base form. The goal of stemming is to reduce words to their core meaning, and to group together words that have the same root, regardless of their inflection.

For example, the words "jumping," "jumps," and "jumped" would all be reduced to the stem "jump." Similarly, "running," "runner," and "ran" would be reduced to the stem "run." This can be useful for tasks such as text classification and information retrieval, as it can help to reduce the dimensionality of the data by reducing the number of unique words.

There are different algorithms that can be used for stemming, such as the Porter[13] stemmer, Snowball[14] stemmer, and the Lancaster[15] stemmer. Each algorithm has its own strengths and weaknesses, and the choice of algorithm will depend on the specific application and the language being used. But unfortunately, none of them can be used in Greek and more specifically for complex technical problems such as patents.

After analysing the glossary problem and after many attempts it was concluded that the best way to stem the corpus and not lose some critical words was by removing the final «ς», «υ».

3.3.3 Lemmatization

Lemmatization is a process in natural language processing (NLP) that reduces words to their base or root form. This is done by identifying the word's lemma, or the base form of the word,

[12] https://www.nltk.org/ Is a python library with predefined well known stopwords.

[13] Porter Stemming Algorithm (tartarus.org)

[14] Snowball (snowballstem.org)

[15] Another stemmer | ACM SIGIR Forum

and mapping it to the word. Lemmatization is similar to stemming, but it results in an existing word, whereas stemming may produce a non-existing word. This can be useful for tasks such as text classification and information retrieval, as it can help to reduce the size of the vocabulary by reducing the number of unique words. A tool that was used was spaCy7. Unfortunately, it was not performing well for technical documents, and an inhouse lemmatization was developed library which outperformed spaCy even with a small number of words. This is currently under development.

3.3.4 Tokenizer

Tokenization is the process of breaking down a larger piece of text into smaller units called tokens. These tokens can be words, phrases, sentences, or paragraphs, depending on the task and the level of granularity needed. Tokenization is a fundamental step in many natural language processing (NLP) tasks, such as text classification, information retrieval, and language translation. It is used to divide a text into meaningful segments, making it easier to analyze and work with the data. For tokenization of the dataset Keras8 framework was used and then Bag of Words was created upon our final corpus.

3.3.5 Embeddings

For the creation of embeddings, Keras' embedding layer was used. This layer is used as part of the deep learning models where the embeddings are learned along with the model itself. Each model has their own embeddings. A size of 128 was chosen for all of our models.

3.3.6 Evaluation Methods

In neural networks, the train-validation split is a technique used to divide the data into two sets: the training set and the validation set. The training set is used to train the model, while the validation set is used to evaluate the model's performance during the training process. The training set is used to fit the model's weights and biases to the data, while the validation set is used to monitor the model's performance during the training process and select the best hyperparameters. The goal is to find the best set of weights and biases that minimize the error on the validation set.

Moreover, he train-validation split is an important technique in neural networks as it allows to detect overfitting. Overfitting occurs when a model is too complex, and it performs well on the training set but poorly on the validation set. By using the validation set, we can detect overfitting by monitoring the performance of the model on unseen data and adjust the model's complexity accordingly. The split ratio between train and validation sets is usually of 80% - 20% but it can vary depending on the size of the dataset. In some cases, a separate test set is also used, to evaluate the model's performance on unseen data once it has been trained. It is a technique used in neural networks to divide the data into two sets: the training set and the validation set. The training set is used to train the model, while the validation set is used to evaluate the model's performance during the training process. This allows for detecting overfitting and adjusting the model's complexity accordingly. [30]

For assessing the model's performance on performed tasks it was used a set of metrics. The goal is to determine how well the model is able to make predictions on unseen data. Metrics used were Accuracy, F1 score, Precision and Recall.

Accuracy is a commonly used metric in machine learning to evaluate the performance of a model on a classification task. It represents the proportion of correctly classified instances out of the total number of instances in the dataset. Basically, it is the number of correctly classified instances divided total number of instances. It ranges from 0 to 1, where 1 is perfect accuracy and 0 is no accuracy. [33]

Precision is the number of true positive predictions divided by the total number of true positive and false positive predictions. It is a measure of how many of the positive predictions made by the model are correct. [33]

Recall is the number of true positive predictions divided by the total number of true positive and false negative predictions. It is a measure of how many of the actual positive cases the model correctly identified. [33]

The *F1 score* is a balance between precision and recall. It is the harmonic mean of the two, which means that it gives more weight to low values. The F1 score ranges between 0 and 1, with a score of 1 indicating perfect precision and recall, and a score of 0 indicating that the model made no correct predictions.

3.4 Methodology

After understanding the contribution of deep learning in such concepts, it was clear that such approaches could solve the problem of classifying patents. The depth of how much of this complex classification problem could be solved with such approaches is yet to be discovered.

3.4.1 Dense Neural Networks

The first model evaluated was a Dense neural network also known as fully connected (FC) network. A simple structure of NN was used with fully connected layers and each one followed by ReLU [36] activation function. In total 5 layers were used with the last one being the classification layer followed by a softmax activation function. The Adam optimizer was selected using categorical cross entropy as loss function.

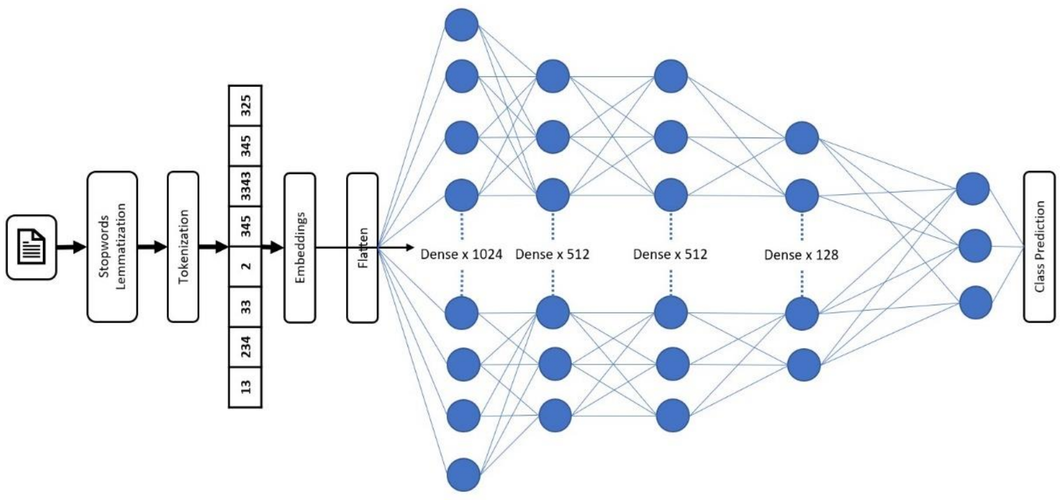


Figure 3.2: Dense Neural Network Architecture.

3.4.2 Convolutional Neural Networks

After careful consideration of the related work, another chosen methodology was that of Convolutions Neural Networks (CNN). In this implementation two 1-dimensional convolutional layers followed by ReLU activation functions were used, having 256 and 128 filter shape, respectively. Afterwards the processed data were flattened to feed dense layers and a final classification layer, while Adam optimizer was selected having categorical cross entropy as loss function.

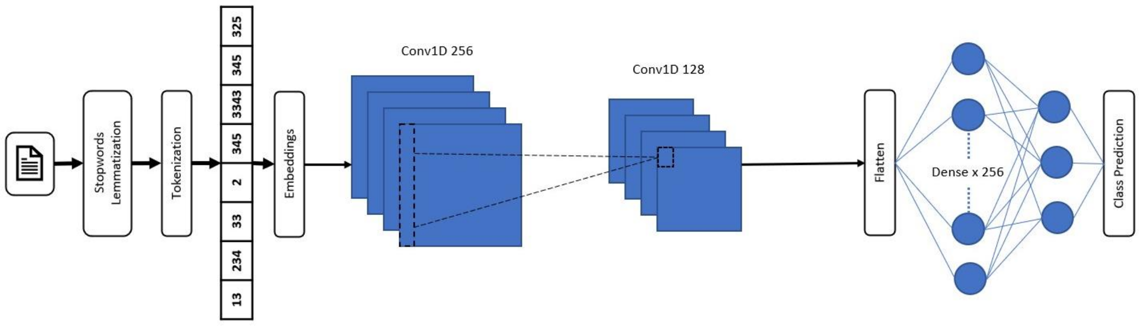


Figure 3.3: Convolutional Neural Network Architecture.

3.4.3 Recurrent Neural Networks

Recurrent neural networks were another methodology chosen to be investigated. In recent literature review, recurrent neural networks have amazing results in NLP tasks. They can grasp the sequence of data, which is crucial to perceive the input pieces of text, since the order of words play an important role in the semantic meaning. Moreover, in this comparison the use of Long Short-Term Memory module was selected with size of 64 neurons and a dropout of 0.1 (Figure 3.4). After the LSTM layers dense layers were added with the final layer being the classification layer and Adam optimizer was selected optimized using categorical cross entropy as loss function.

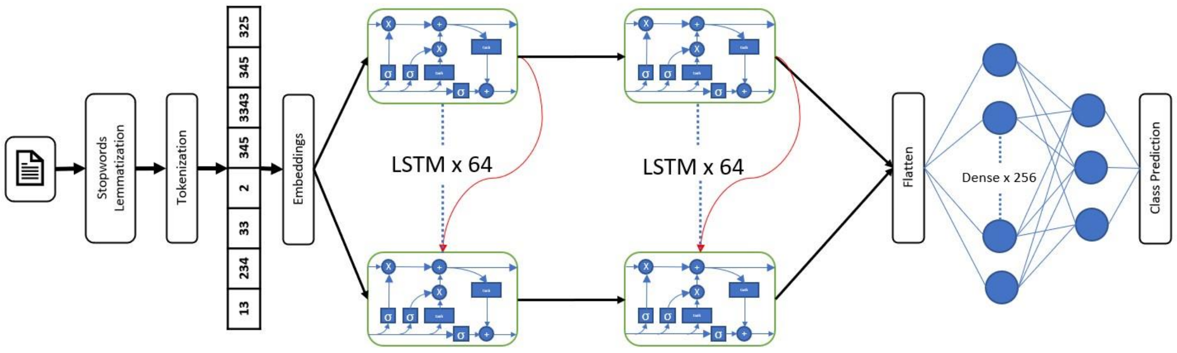


Figure 3.4: Recurrent Neural Network Architecture (LSTM).

3.4.4 Transformers

Transformers [3], such as ALBERT [44] and RoBERTa [42], are considered to be the state-of-the-art in the NLP field. Thus, fine-tuning of Greek-BERT was examined, which outperforms other multilingual models [11] when it comes to Greek text classification, on the collected patent classification dataset. Greek-BERT has the same architecture with the English-base-uncased 9 (12-layer, 768-hidden, 12-heads, 110M parameters, as shown in Figure 3.5 BERT model [15], trained on Greek Wikipedia10, European Parliament Proceedings Parallel Corpus 11and a Greek part of OSCAR12. A hyper-tuning was done for learning-rate (LR) and then the results of Greek-BERT were compared with the results of XLM [41] which is a multilingual transformer-based model.

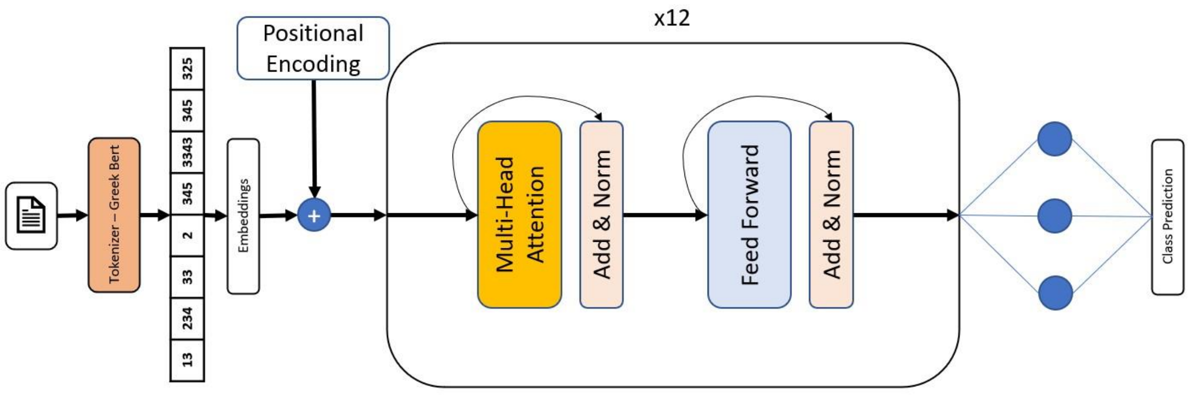


Figure 3.5: Greek-BERT Architecture. [11]

4 Results

In this chapter, all the result are presented and discussed. In each trial, several plots will be presented such as loss per epoch, accuracy per epoch and also a confusion matrix indicating how many data from test subset were classified correctly and how many were misclassified for another class, providing more intuition. Finally, a classification report is presented which presents precision, recall, F1-score and how many documents from the test dataset they were used for evaluation.

4.1 Direct Evaluation on the Greek Patent Dataset

As abovementioned, in this development, dense neural networks were used. The training was performed in the Greek dataset, and more particular to the 3 classes set that was created, then in the 8 class and a comparison of the results is presented.

4.1.1 Vanilla Neural Nets

For the 3 classes problem the accuracy that was achieved was 88%. This was a really good result for dense neural networks, taking into consideration that dense neural networks, according to relevant research, don't perform that well in such problems. The following figures present the obtained accuracy and loss (Figure 4.1), the confusion matrix (Figure 4.2) and the corresponding classification report (Table 4.1).

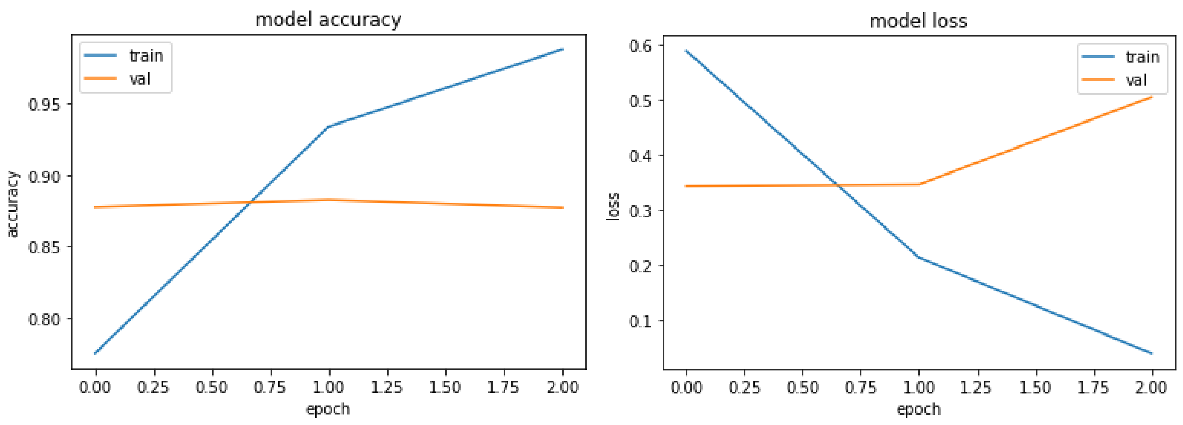


Figure 4.1 Accuracy and Loss graph for FCNN on the GPD-3.

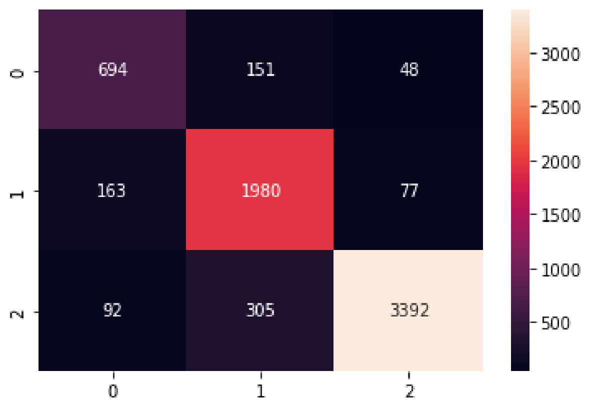
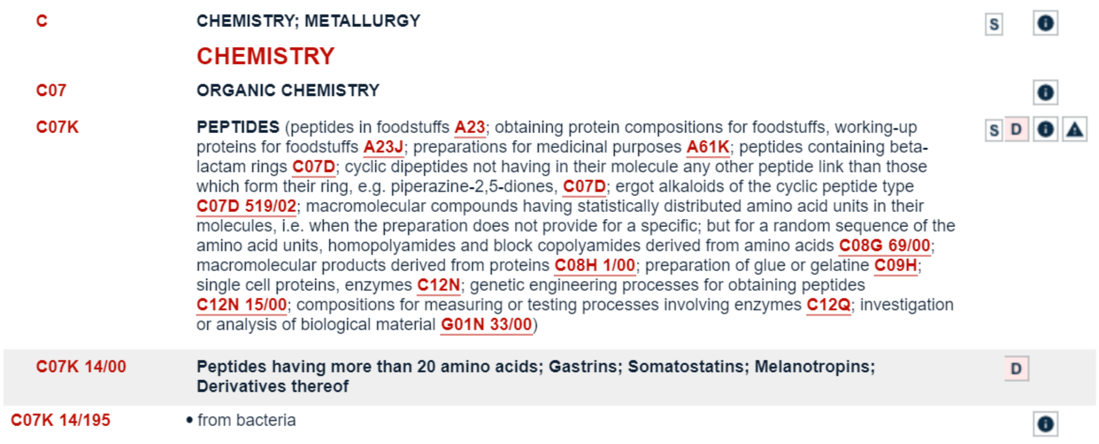


Figure 4.2: Confusion matrix for the FC NN on the GPD-3.

Table 4.1: Classification Report for the FC NN on the GPD-3.

	Precision	Recall	F1-score	Support
ΗΛΕΚΤΡΟΛΟΓΙΚΟ	0.71	0.82	0.76	875
ΜΗΧΑΝΟΛΟΓΙΚΟ	0.84	0.88	0.86	2,281
ΧΗΜΙΚΟ	0.96	0.90	0.93	3,746
ACCURACY	-	-	0.88	6,902
MACRO AVG	0.84	0.87	0.85	6,902
WEIGHTED AVG	0.89	0.88	0.89	6,902

For the 8 classes problem, apparently the accuracy was significantly lower. Dense neural networks manage to get a 67% accuracy (Figure 4.5, Table 4.2). We can clearly see a big confusion in classes A and C which is the hardest classes to be predicted (Figure 4.6). This is because they have a lot of similar words that are overlapping between each class. For example, the classification of C07K14/195 as seen in Figure 4.3, refers to peptides in general as it is part of chemistry.



C CHEMISTRY; METALLURGY

CHEMISTRY

C07 ORGANIC CHEMISTRY

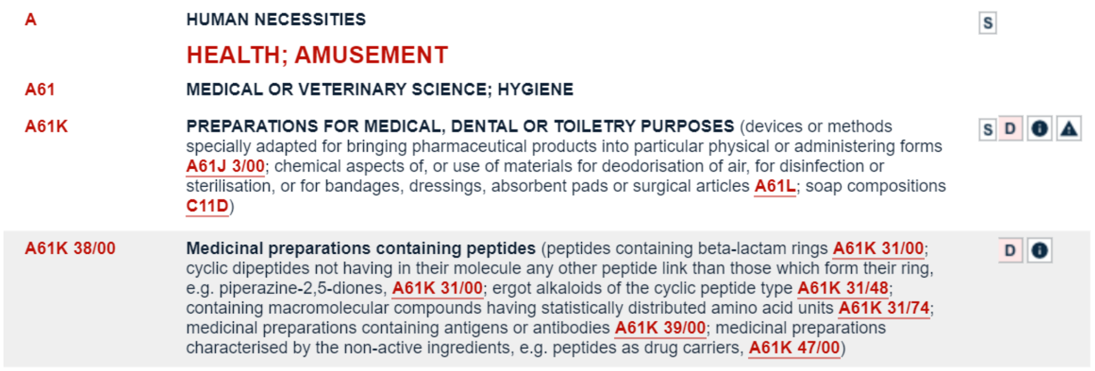
C07K **PEPTIDES** (peptides in foodstuffs A23; obtaining protein compositions for foodstuffs, working-up proteins for foodstuffs A23J; preparations for medicinal purposes A61K; peptides containing beta-lactam rings C07D; cyclic dipeptides not having in their molecule any other peptide link than those which form their ring, e.g. piperazine-2,5-diones, C07D; ergot alkaloids of the cyclic peptide type C07D 519/02; macromolecular compounds having statistically distributed amino acid units in their molecules, i.e. when the preparation does not provide for a specific; but for a random sequence of the amino acid units, homopolyamides and block copolyamides derived from amino acids C08G 69/00; macromolecular products derived from proteins C08H 1/00; preparation of glue or gelatine C09H; single cell proteins, enzymes C12N; genetic engineering processes for obtaining peptides C12N 15/00; compositions for measuring or testing processes involving enzymes C12Q; investigation or analysis of biological material G01N 33/00)

C07K 14/00 **Peptides having more than 20 amino acids; Gastrins; Somatostatins; Melanotropins; Derivatives thereof**

C07K 14/195 • from bacteria

Figure 4.3: Example of IPC showing description of C07K14/195.

Also, the classification A61K38/00 as seen in Figure 4.4, refers to medicinal preparations containing peptides. This creates a big overlap in the terminology used in both IPCs because they have a lot of common words due to the similarity of the content but yet again, they are different.



A HUMAN NECESSITIES

HEALTH; AMUSEMENT

A61 MEDICAL OR VETERINARY SCIENCE; HYGIENE

A61K **PREPARATIONS FOR MEDICAL, DENTAL OR TOILETRY PURPOSES** (devices or methods specially adapted for bringing pharmaceutical products into particular physical or administering forms A61J 3/00; chemical aspects of, or use of materials for deodorisation of air, for disinfection or sterilisation, or for bandages, dressings, absorbent pads or surgical articles A61L; soap compositions C11D)

A61K 38/00 **Medicinal preparations containing peptides** (peptides containing beta-lactam rings A61K 31/00; cyclic dipeptides not having in their molecule any other peptide link than those which form their ring, e.g. piperazine-2,5-diones, A61K 31/00; ergot alkaloids of the cyclic peptide type A61K 31/48; containing macromolecular compounds having statistically distributed amino acid units A61K 31/74; medicinal preparations containing antigens or antibodies A61K 39/00; medicinal preparations characterised by the non-active ingredients, e.g. peptides as drug carriers, A61K 47/00)

Figure 4.4: Example of IPC showing description of A61K38/00.

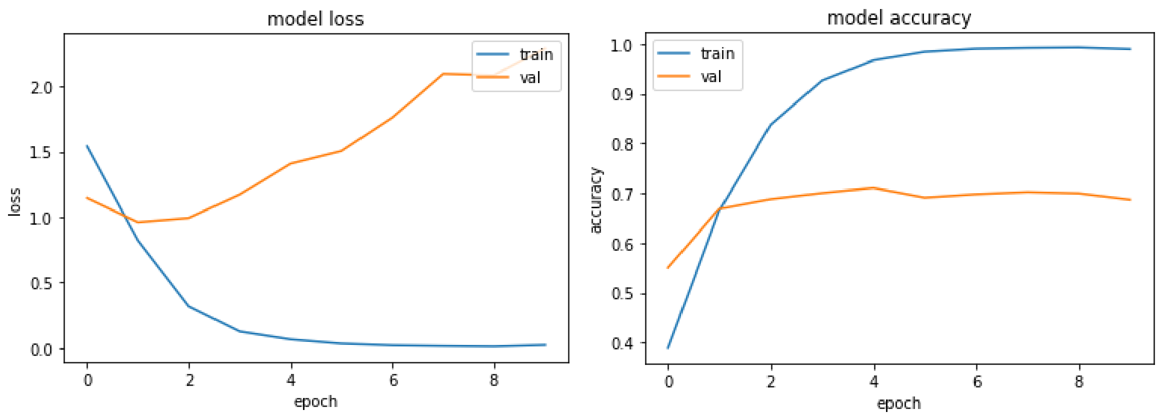
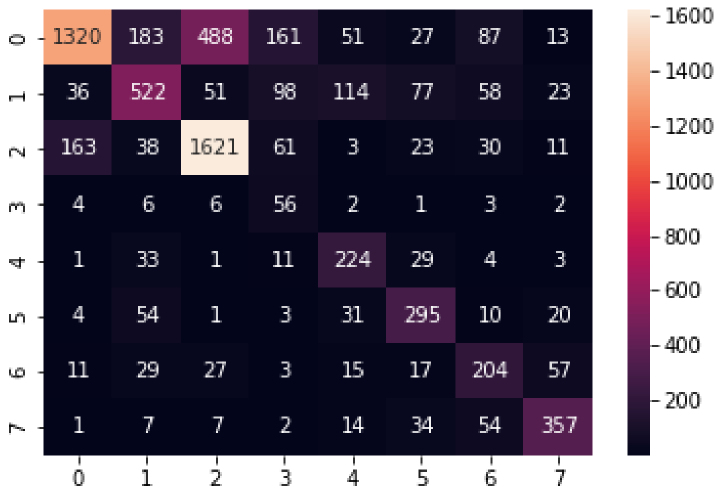


Figure 4.5: Accuracy and Loss graph for FCNN on the GPD-8.



	0	1	2	3	4	5	6	7
0	1320	183	488	161	51	27	87	13
1	36	522	51	98	114	77	58	23
2	163	38	1621	61	3	23	30	11
3	4	6	6	56	2	1	3	2
4	1	33	1	11	224	29	4	3
5	4	54	1	3	31	295	10	20
6	11	29	27	3	15	17	204	57
7	1	7	7	2	14	34	54	357

Figure 4.6: Confusion matrix for the FC NN on the GPD-8.

Table 4.2: Classification Report for the FCNN on the GPD-8.

	Precision	Recall	F1-score	Support
A	0.86	0.57	0.68	2,330
B	0.60	0.53	0.56	979
C	0.74	0.83	0.78	1,950
D	0.14	0.70	0.24	80
E	0.49	0.73	0.59	306
F	0.59	0.71	0.64	418
G	0.45	0.56	0.50	363
H	0.73	0.75	0.74	476
ACCURACY	-	-	0.67	6,902
MACRO AVG	0.58	0.67	0.59	6,902
WEIGHTED AVG	0.72	0.67	0.68	6,902

4.1.2 Convolutional Neural Networks

Another method that has been chosen was CNNs (structure se being described in previous section 3.5.2). According to the related works, CNNs performs well in NLP problems and are able to distinguish features in matrices better than fully connected dense neural networks. Unfortunately, this was not the case exactly. For the 3 classes trial, CNNs performed similarly to dense neural networks. The following figures present the obtained accuracy and loss (Figure 4.7), the confusion matrix (Figure 4.8) and the corresponding classification report (Table 4.3).

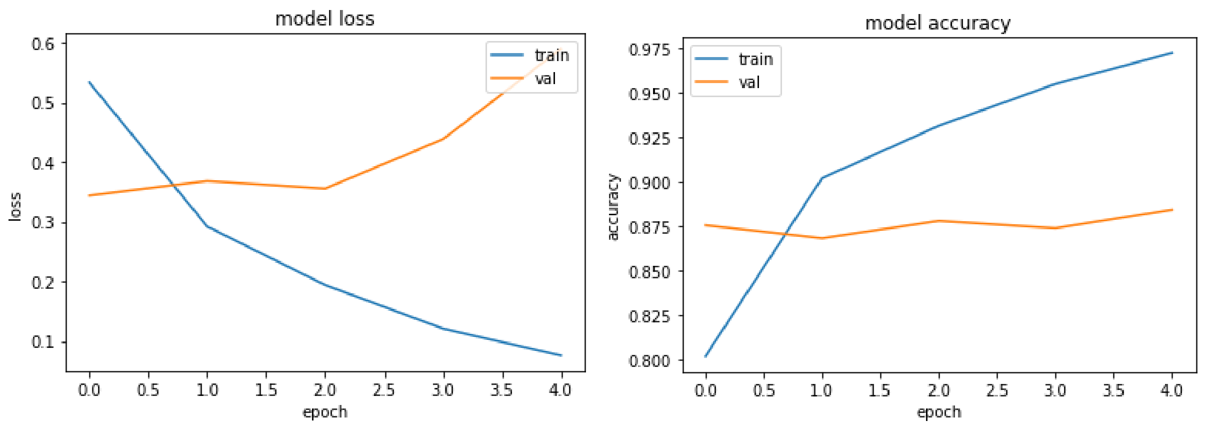


Figure 4.7: Accuracy and Loss graph for CNN on the GPD-3.

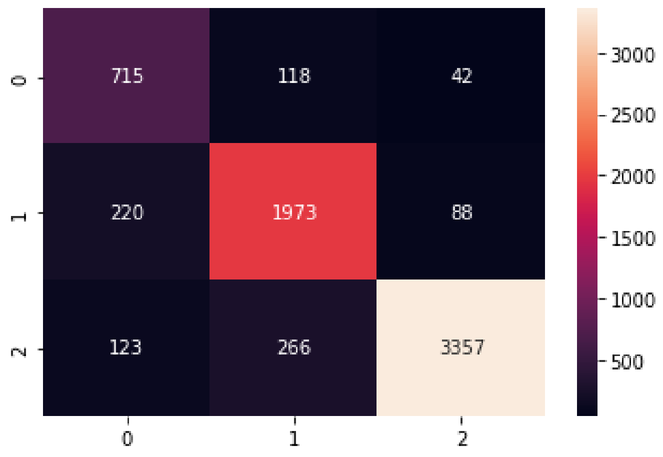


Figure 4.8: Confusion matrix for the CNN on the GPD-3.

Table 4.3: Classification Report for the CNN on the GPD-3.

	Precision	Recall	F1-score	Support
ΗΛΕΚΤΡΟΛΟΓΙΚΟ	0.68	0.82	0.74	875
ΜΗΧΑΝΟΛΟΓΙΚΟ	0.84	0.86	0.85	2,281
ΧΗΜΙΚΟ	0.96	0.90	0.93	3,746
ACCURACY	-	-	0.88	6,902
MACRO AVG	0.83	0.86	0.84	6,902
WEIGHTED AVG	0.88	0.88	0.88	6,902

Similarly, as for the GPD-3 results, also for the GPD-8, it is clear that they have similar results to the FC neural network's. This means that CNN are a little bit unable to grasp extra features than the FC neural networks. The following figures present the obtained accuracy and loss (Figure 4.9), the confusion matrix (Figure 4.10) and the corresponding classification report (Table 4.4).

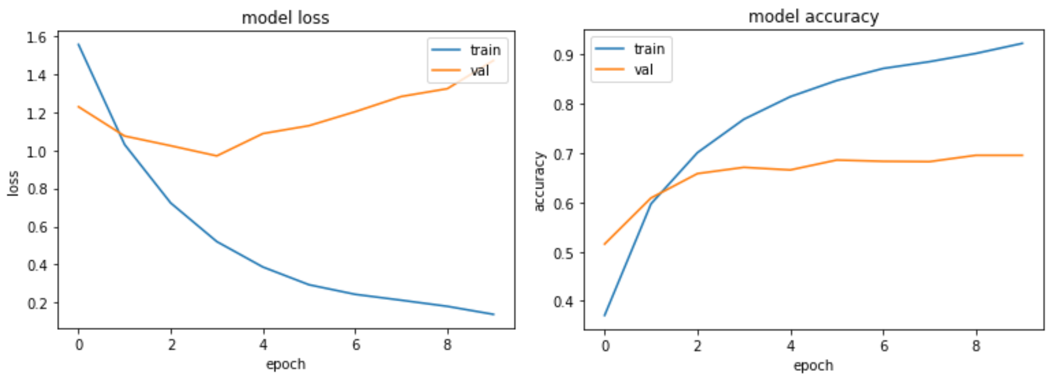


Figure 4.9: Accuracy and Loss graph for CNN in 8 Classes problem.

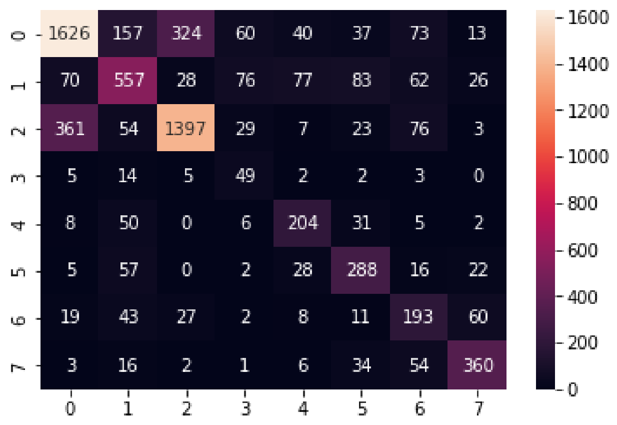


Figure 4.10: Confusion matrix for the CNN on the GPD-8.

Table 4.4: Classification Report for the CNN on the GPD-8.

	Precision	Recall	F1-score	Support
A	0.78	0.70	0.73	2,330
B	0.59	0.57	0.58	979
C	0.78	0.72	0.75	1,950
D	0.22	0.61	0.32	80
E	0.55	0.67	0.60	306
F	0.57	0.69	0.62	418
G	0.40	0.53	0.46	363
H	0.74	0.76	0.75	476
ACCURACY	-	-	0.68	6,902
MACRO AVG	0.58	0.65	0.60	6,902
WEIGHTED AVG	0.70	0.68	0.69	6,902

4.1.3 LSTM

The next methodology that was implemented was that of LSTM neural networks; they are commonly used in NLP tasks due to their ability to capture long-term dependencies in sequential data. Therefore, in this research a hyperparameter tuning tool was used on this type of NNs to get the best hyper parameters which can be seen in Appendix A. The results were similar to the other two previous methods (Figure 4.11, Figure 4.12, Table 4.5).

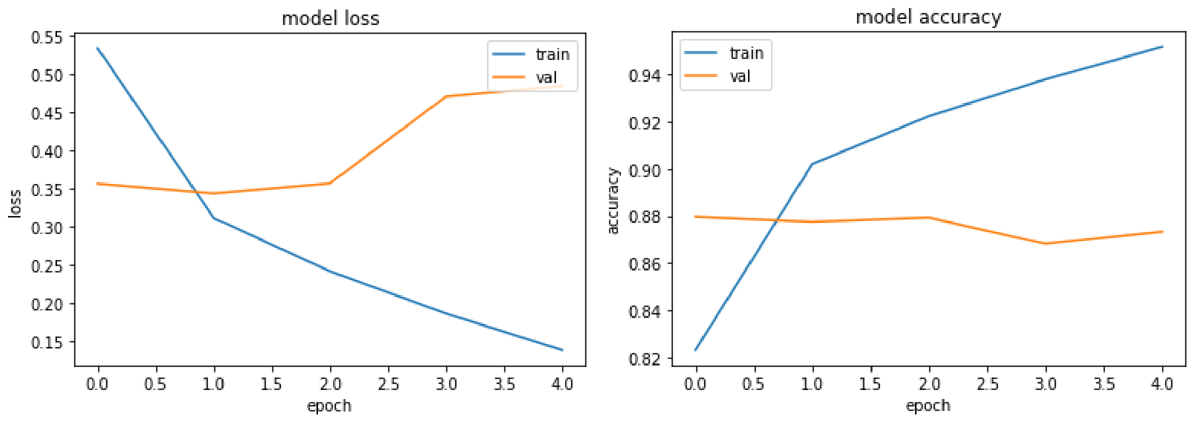


Figure 4.11: Accuracy and Loss graph for LSTM on the GPD-3.

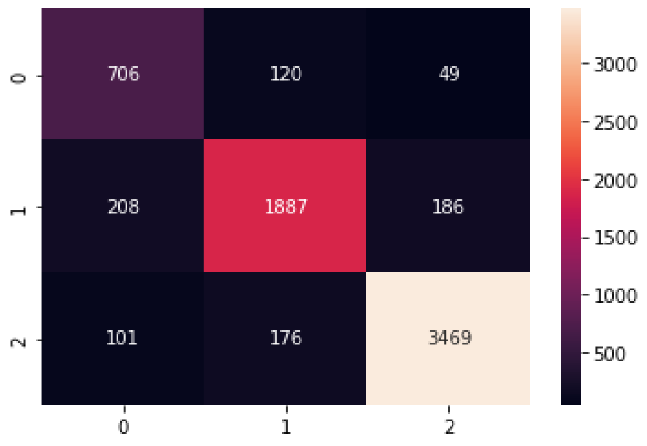


Figure 4.12: Confusion matrix for the LSTM on the GPD-3.

Table 4.5: Classification Report for the LSTM on the GPD-3.

	Precision	Recall	F1-score	Support
ΗΛΕΚΤΡΟΛΟΓΙΚΟ	0.70	0.81	0.75	875
ΜΗΧΑΝΟΛΟΓΙΚΟ	0.86	0.83	0.85	2,281
ΧΗΜΙΚΟ	0.94	0.93	0.93	3,746
ACCURACY	-	-	0.88	6,902
MACRO AVG	0.83	0.85	0.84	6,902
WEIGHTED AVG	0.88	0.88	0.88	6,902

For the 8-class problem the results were similar to the other models also. Although, there was a small drop by 1% the results are almost similar as presented in Figure 4.13, Figure 4.14 and Table 4.6.

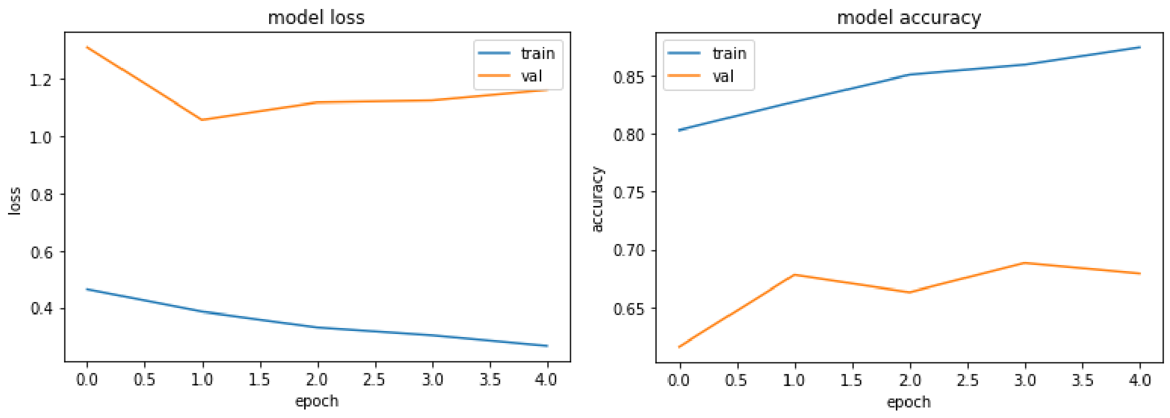


Figure 4.13: Accuracy and Loss graph for LSTM on the GPD-8.

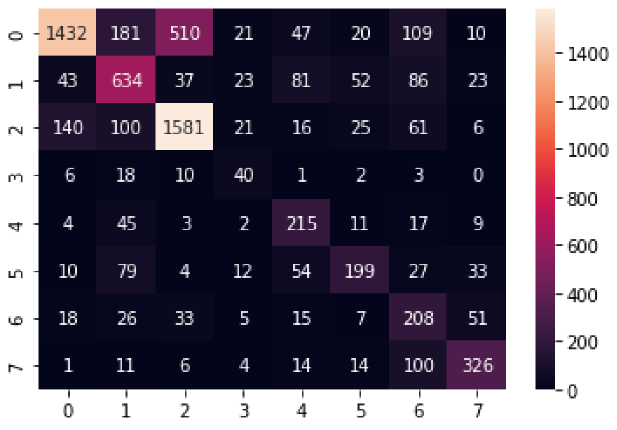


Figure 4.14: Confusion matrix for the LSTM on the GPD-8.

Table 4.6: Classification Report for the LSTM on the GPD-8.

	Precision	Recall	F1-score	Support
A	0.87	0.61	0.72	2,330
B	0.58	0.65	0.61	979
C	0.72	0.81	0.76	1,950
D	0.31	0.50	0.38	80
E	0.49	0.70	0.57	306
F	0.60	0.48	0.53	418
G	0.34	0.57	0.43	363
H	0.71	0.68	0.70	476
ACCURACY	-	-	0.67	6,902
MACRO AVG	0.58	0.63	0.59	6,902
WEIGHTED AVG	0.71	0.67	0.68	6,902

4.1.4 Transformers

4.1.4.1 XLM

The first trial of transformer-based networks was with multilingual XLM model. As aforementioned, XLM is a multilingual language model developed by Facebook AI. It was trained on a massive corpus of text data from a wide range of languages, allowing it to perform well on tasks such as translation and text classification across many languages. XLM can handle over 100 languages and has demonstrated impressive results in cross-lingual transfer learning. Although, it seems a suitable candidate for the presented task, the validation accuracy that was able to achieve for the GPD-3 was about 55% on the Greek test dataset with a validation loss of 0.9576, which probably can be explained by the highly technical terms presented in our corpus. Since we had so low validation accuracy with the GPD-3 we would certainly have less accuracy with the GPD-8, therefore since it takes so much time to train, these results will not be researched.

4.1.4.2 Greek-Bert

As aforementioned, Greek-BERT is a version of the popular language model BERT that is pre-trained on Greek language text. BERT is a deep learning model that can be fine-tuned for various natural language processing tasks such as text classification and named entity recognition. Not surprisingly it outperformed any other method used. It was able to get a 91% of validation accuracy in the 3-class problem and an 82% validation accuracy for the 8-class problem (Figure 4.15, Table 4.7). The only hyper tuning that was done with the model, was to get the best learning rate which was equal to 0.0001. The results were:

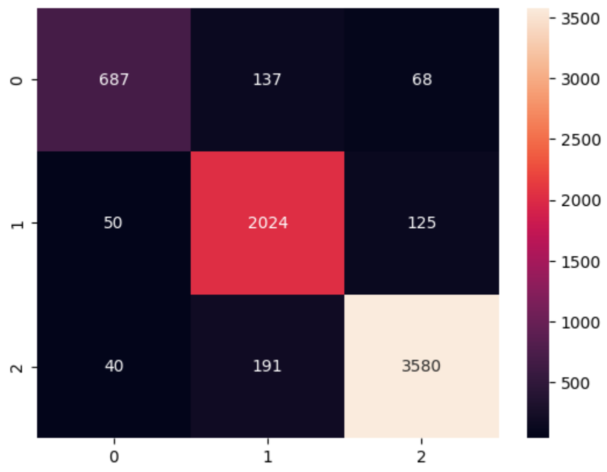


Figure 4.15: Confusion matrix for the Greek-BERT on the GPD-3.

Table 4.7: Classification Report for the Greek-BERT on the GPD-3.

	Precision	Recall	F1-score	Support
ΗΛΕΚΤΡΟΛΟΓΙΚΟ	0.88	0.77	0.82	892
ΜΗΧΑΝΟΛΟΓΙΚΟ	0.86	0.92	0.89	2,199
ΧΗΜΙΚΟ	0.95	0.94	0.94	3811
ACCURACY	-	-	0.91	6,902
MACRO AVG	0.90	0.88	0.89	6,902
WEIGHTED AVG	0.91	0.91	0.91	6,902

For GPD-8, it is clear that it minimizes a lot the confusion that was observed between classes A and C (Figure 4.16). The F1-score achieved on the GPD-8 is also the highest (Table 4.8).

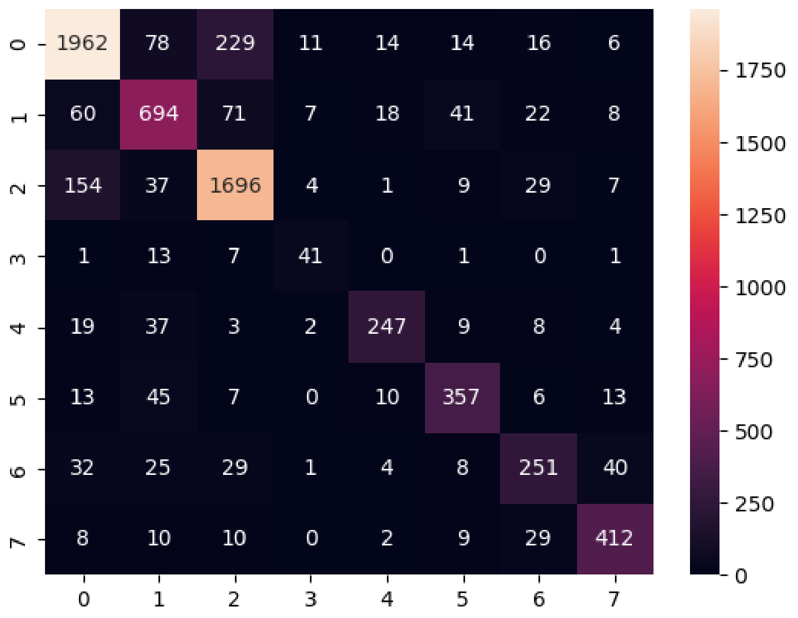


Figure 4.16: Confusion matrix for the Greek-BERT on the GPD-8.

Table 4.8: Classification Report for the Greek-BERT on the GPD-8.

	Precision	Recall	F1-score	Support
A	0.87	0.84	0.86	2,330
B	0.74	0.75	0.75	921
C	0.83	0.88	0.85	1,937
D	0.62	0.64	0.63	64
E	0.83	0.75	0.79	329
F	0.80	0.79	0.79	451
G	0.70	0.64	0.67	390
H	0.84	0.86	0.85	480
ACCURACY	-	-	0.82	6,902
MACRO AVG	0.78	0.77	0.77	6,902
WEIGHTED AVG	0.82	0.82	0.82	6,902

4.1.5 Discussion

For the 3-GPD after comparing the classifications reports as seen in Table 4.9, and more specifically the F1-scores of each model, it can be seen that the best model was the Greek-BERT. Although Greek-BERT is not trained upon patents the results that it was able to achieve was by far more after only 2 epochs on the fine-tuning process.

Table 4.9: Comparison table of F1-scores on the GPD-3 results.

	FCNN	CNN	LSTM	Greek-BERT	Support
ΗΛΕΚΤΡΟΛΟΓΙΚΟ	0.76	0.74	0.77	**0.82**	875
ΜΗΧΑΝΟΛΟΓΙΚΟ	0.86	0.85	0.86	**0.89**	2,281
ΧΗΜΙΚΟ	0.93	0.93	0.93	**0.94**	3,746
ACCURACY	0.88	0.88	0.89	**0.91**	6,902
MACRO AVG	0.85	0.84	0.85	**0.89**	6,902
WEIGHTED AVG	0.89	0.88	0.89	**0.91**	6,902

Similarly, after comparing the F1-scores classification tables (Table 4.10) for the GPD-8, it can be seen that the accuracy difference is much greater than the GPD-3. Greek-BERT outperformed all the other models and gave much more better results in a more complex classification problem.

Table 4.10: Comparison table of F1-scores on the GPD-8 results.

	FCNN	CNN	LSTM	Greek-BERT	Support
A	0.68	0.73	0.72	0.86	2,330
B	0.56	0.58	0.61	0.75	979
C	0.78	0.75	0.76	0.85	1,950
D	0.24	0.32	0.38	0.63	80
E	0.59	0.60	0.57	0.79	306
F	0.64	0.62	0.53	0.79	418
G	0.50	0.46	0.43	0.67	363
H	0.74	0.75	0.70	0.85	476
ACCURACY	0.67	0.68	0.67	**0.82**	6,902
MACRO AVG	0.59	0.60	0.59	0.77	6,902
WEIGHTED AVG	0.68	0.69	0.68	0.82	6,902

4.2 English Patent Dataset

4.2.1.1 GLOVE-based LSTM

In these trials, GLOVE was used as pre-trained embeddings. GLOVE (Global Vectors for Word Representation) is a pre-trained word embedding model that represents words as high-dimensional vectors. It is trained on a large corpus of text to capture the contextual relationship between words. GLOVE has been widely used in many NLP tasks, such as text classification, language translation, and sentiment analysis, as it provides an efficient and effective way to encode linguistic information into a numerical format.

An LSTM model was used with 2 LSTM layers with size of 64 and after a flatten of data 2 fully connected layers was used with 256 and 128 layers respectively.

The dataset here is 1,317,424 English patents, because there were some limitations due to limited RAM that it could be used.

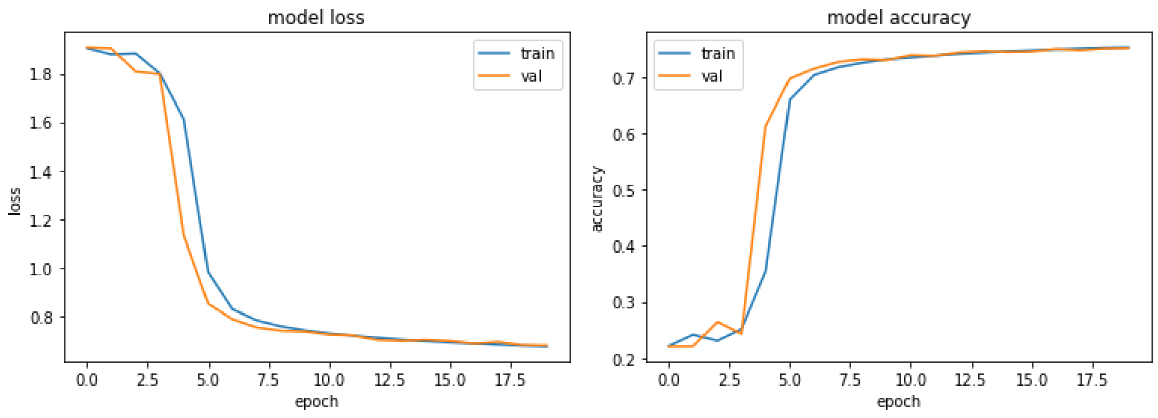
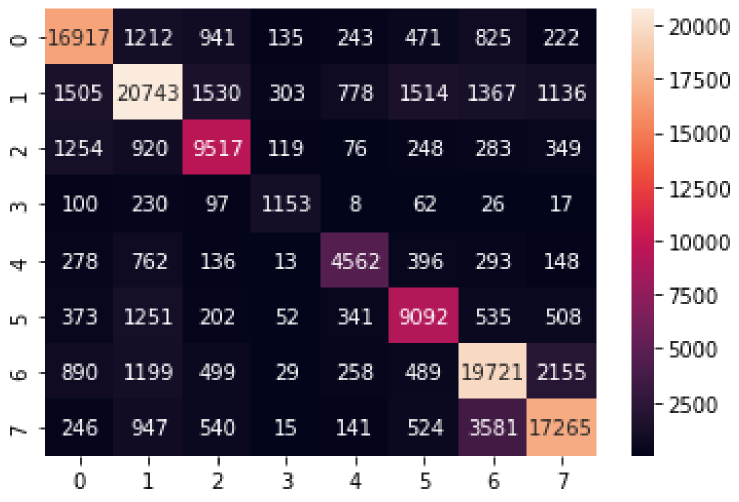


Figure 4.17: Accuracy and Loss graph for GLOVE on the EPD-8.



	0	1	2	3	4	5	6	7
0	16917	1212	941	135	243	471	825	222
1	1505	20743	1530	303	778	1514	1367	1136
2	1254	920	9517	119	76	248	283	349
3	100	230	97	1153	8	62	26	17
4	278	762	136	13	4562	396	293	148
5	373	1251	202	52	341	9092	535	508
6	890	1199	499	29	258	489	19721	2155
7	246	947	540	15	141	524	3581	17265

Figure 4.18: Confusion matrix for the GLOVE on the EPD-8.

Table 4.11: Classification Report for the GLOVE on the EPD-8.

	Precision	Recall	F1-score	Support
A	0.78	0.81	0.80	20,966
B	0.76	0.72	0.74	28,876
C	0.71	0.75	0.73	12,766
D	0.63	0.68	0.66	1,693
E	0.71	0.69	0.70	6,588
F	0.71	0.74	0.72	12,354
G	0.74	0.78	0.76	25,240
H	0.79	0.74	0.77	23,259
ACCURACY	-	-	0.75	131,742
MACRO AVG	0.73	0.74	0.73	131,742
WEIGHTED AVG	0.75	0.75	0.73	131,742

4.2.2 LSTM

Because LSTM are the most widely used, it was the only method that was examined in training with the English dataset. In the English language, there are a lot of methods that can outperform neural networks that are being trained from scratch. Transformers, pre-trained embeddings etc., are methods that can give high accuracies with small number of iterations(epochs). For the 8-class model that it was trained, we got a 72% of validation accuracy which is better than the previous trials but not so good for the big dataset that we had.

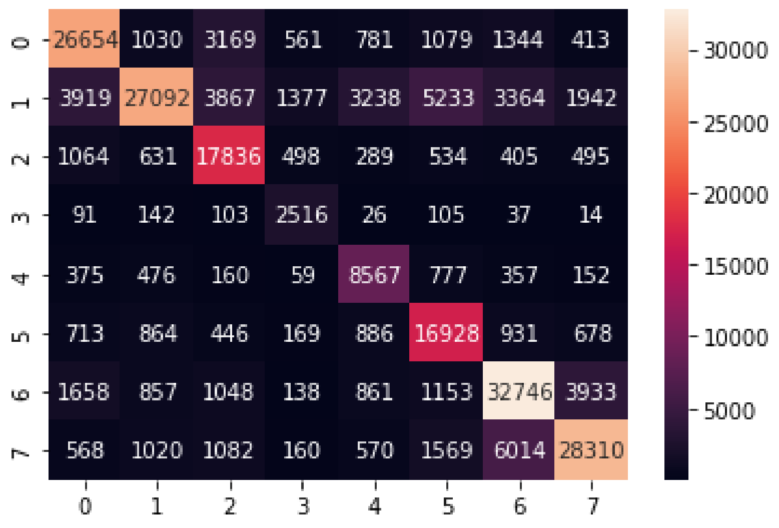


Figure 4.19: Confusion matrix for the LSTM on the EPD-8.

Table 4.12: Classification Report for the LSTM on the EPD-8.

	Precision	Recall	F1-score	Support
A	0.76	0.76	0.76	35,031
B	0.84	0.54	0.66	50,032
C	0.64	0.82	0.72	21,752
D	0.46	0.83	0.59	3,034
E	0.56	0.78	0.66	10,923
F	0.62	0.78	0.59	21,615
G	0.72	0.77	0.75	42,394
H	0.79	0.72	0.75	93,293
ACCURACY	-	-	0.72	224,074
MACRO AVG	0.68	0.75	0.70	224,074
WEIGHTED AVG	0.74	0.72	0.72	224,074

4.2.3 English RoBERTa

To compare another transformer-based model against the GLOVE pre-trained embeddings, we used RoBERTa. RoBERTa [42] (Robustly Optimized BERT Pre-training Approach) is a variant of the popular BERT language model that was developed by Facebook AI. It is a deep learning model for natural language processing (NLP) tasks, such as text classification, named entity recognition, and question answering. RoBERTa is designed to be more robust and scalable than its predecessor BERT and was pre-trained on a massive corpus of text data from the web. The "RoBERTa base" refers to the standard, uncased version of the RoBERTa model, which has a certain number of parameters and was trained on a specific dataset. After fine-tuning the model with our patent dataset, it was able to achieve 81% of accuracy as seen in

Table 4.13.

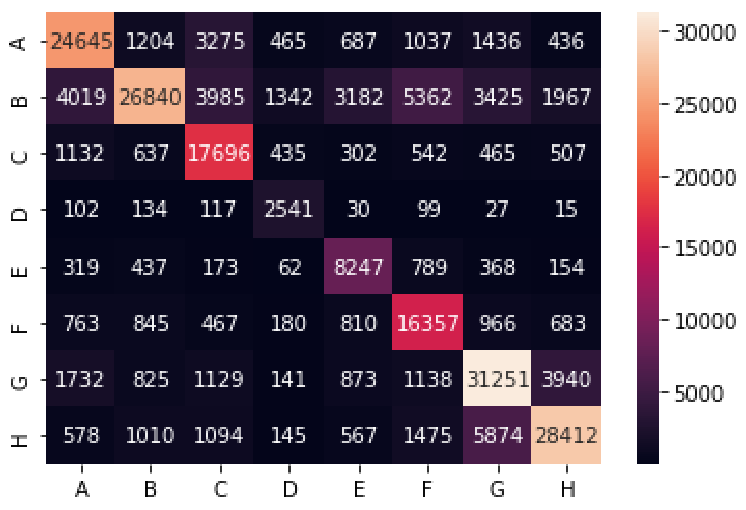


Figure 4.20: Confusion matrix for the RoBERTa fine-tuned on the EPD-8.

Table 4.13: Classification Report for the RoBERTa on the EPD-8.

	Precision	Recall	F1-score	Support
A	0.88	0.85	0.86	35,031
B	0.75	0.73	0.73	50,032
C	0.80	0.87	0.86	21,752
D	0.64	0.65	0.65	3,034
E	0.84	0.74	0.76	10,923
F	0.79	0.80	0.79	21,615
G	0.71	0.63	0.66	42,394
H	0.83	0.84	0.84	93,293
ACCURACY	-	-	0.81	224,074
MACRO AVG	0.77	0.76	0.76	224,074
WEIGHTED AVG	0.81	0.81	0.81	224,074

4.2.4 Discussion

For the EPD-8 dataset a comparison is shown in **Error! Reference source not found.**. Transformer-based model (RoBERTa) outperformed all the other methods and also the pre-trained embeddings GLOVE.

Table 4.14: Comparison table of F1-scores on the EPD-8 results and validated of GPVD Translated to English. (T=Translated)

	GLOVE	LSTM	RoBERTa	GPVD (T) on LSTM
A	0.80	0.73	0.86	0.80
B	0.74	0.58	0.73	0.65
C	0.73	0.75	0.86	0.78
D	0.66	0.32	0.65	0.62
E	0.70	0.60	0.76	0.56
F	0.72	0.62	0.79	0.62
G	0.76	0.46	0.66	0.69
H	0.77	0.75	0.84	0.65
ACCURACY	0.75	0.68	0.81	0.75
MACRO AVG	0.73	0.60	0.76	0.67
WEIGHTED AVG	0.73	0.69	0.81	0.75

4.3 Machine Translation for Data Augmentation Purposes

In this trial a subset of the GPD was used, named as Greek Patent Validation Dataset (GPVD-8). Google translate was used to translate Greek patents and classify them with the English model. More specifically, 1,000 random patents were taken from the Greek dataset and were translated with google translator. This was a proof of concept to evaluate whether the machine translated patents had similar results to patents that were originally drafted in a specific language. It is worth mentioning that we selected only 1,000 examples due to pricing policy of the aforementioned API.

It is shown that the fine-tuned English RoBERTa applied to the translated patents misclassified a lot of examples of class C to that of A (Figure 4.21). Lastly, the GPVD-8 was validated with the fine-tuned Greek-BERT before the translation in order to be verified that the results were similar to the test dataset that was used to validate the performance of the trained models in Table 4.15. Unfortunately, the English model had a 4% drop in the macro F1-score (Table 4.15), however, we argue that the results are promising and could lead to increase in the overall performance of the model if this this data augmentation technique was applied to all the Greek documents.

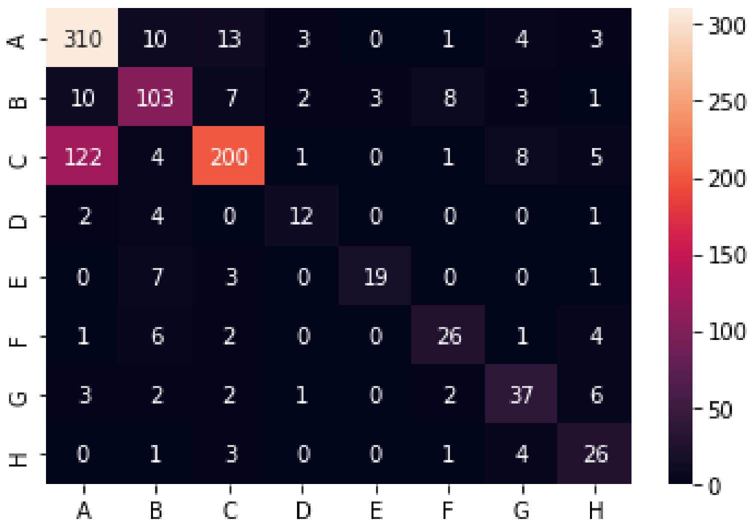


Figure 4.21: Confusion matrix of validation set on the English trained model with Greek translated patents using Google translate library.

Lastly, the GPVD-8 was validated with Greek-BERT before the translation in order to be verified that the results were similar to the test dataset that was used to validate the performance of the trained models in Table 4.15. The results were similar indicating that we could translated them to augment the existing dataset without significantly affecting its performance.

Table 4.15: Comparison table of F1-scores on the GPVD-8 using Greek-BERT and RoBERTa before and after translation, respectively.

	Greek-BERT	RoBERTa
A	0.87	0.78
B	0.72	0.75
C	0.88	0.70
D	0.61	0.63
E	0.77	0.73
F	0.68	0.66
G	0.74	0.77
H	0.68	0.63
ACCURACY	0.83	0.73
MACRO AVG	0.74	0.70
WEIGHTED AVG	0.83	0.73

An example test patent in Greek:

‘Η εφεύρεση σχετίζεται με τους τομείς της βιοτεχνολογίας, της ιολογίας, της επιδημιολογίας και της δημόσιας υγείας, και είναι μέθοδος για την απόκτηση νέου αδρανοποιημένου εμβολίου κατά του κορονοϊού COVID-19. Το βασικό θέμα της εφεύρεσης είναι το στέλεχος του ιού COVID-19 "SARS-CoV-2/KZ_Almaty/04.2020" που απομονώθηκε στην επικράτεια της Δημοκρατίας του Καζακστάν. Το στέλεχος του ιού COVID-19 σύμφωνα με τις βέλτιστες συνθήκες καλλιέργειας παράγεται στο σύστημα κυτταροκαλλιέργειας Vero, αδρανοποιείται με φορμαλδεΰδη, διαυγάζεται με φυγοκέντρηση χαμηλής ταχύτητας, καθαρίζεται και συμπυκνώνεται με διαδιήθηση στη μονάδα διαδιήθησης του συστήματος Millipore Pellicon Cassette. Η αποστείρωση διήθησης πραγματοποιείται μέσω καταρράξεων φίλτρων με διάμετρο πόρων 0,45/0,22 μm. Γέλη υδροξειδίου του αργιλίου 2 % "Algidrogel, 85" προστίθεται στο ληφθέν απόθεμα ιών (ιικό συμπύκνωμα) σε τελική συγκέντρωση 0,5 mg/0,5 ml και εμφιαλώνεται σε γυάλινα φιαλίδια. Το εμβόλιο που λαμβάνεται με αυτόν τον τρόπο είναι ασφαλές κατά την ενδοπεριτοναϊκή εισαγωγή σε λευκά ποντίκια και ενδοφλέβια - σε κουνέλια. Το εμβόλιο παρέχει 80% προστασία έναντι της λοίμωξης COVID-19 για τουλάχιστον 6 μήνες μετά από δύο εμβολιασμούς. Το εμβόλιο διατηρεί τις ιδιότητές του για 12 μήνες στους 4-6°C’.

was translated to the following English text:

The invention relates to the fields of biotechnology, virology, epidemiology and public health, and is method for obtaining of new inactivated vaccine against coronavirus COVID-19. The essence matter of invention is COVID-19 virus "SARS-CoV-2/KZ_Almaty/04.2020" strain isolated on the territory of the Republic of Kazakhstan. The strain of COVID-19 virus according to the optimal cultivation conditions is produced in the Vero cell culture system, inactivated by formaldehyde, clarified by low-speed centrifugation, purified and concentrated by diafiltration-on-diafiltration unit of Millipore Pellicon Cassette system. Sterilizing filtration is carried out through cascades of filters with a pore diameter of 0.45/0.22 μm. 2 % aluminum hydroxide gel "Algidrogel, 85" is added in the obtained virus pool (viral concentrate) to final concentration of 0.5 mg/0.5 ml and bottled in glass vials. The vaccine obtained in this way is safe at intraperitoneal introduction to white mice and intravenously - to rabbits. The vaccine provides 80 % protection against COVID-19 infection for at least 6 months after two vaccinations. The vaccine keeps its properties for 12 months at 4-6°C.

and the model was able to classify the abovementioned example as “ΧΗΜΙΚΟ”.

5 CONCLUSIONS

Patent system is a complex NLP task; it has its own terminologies, unique way of expressing technical concept, it is a fast-paced environment, and it keeps evolving while technology evolves. As an NLP field, there are many problems and areas unexplored which could add a lot of value in many organizations. While these areas are unexplored, deep learning technologies offer a big variety of powerful tools that could be used to solve a numerous of problems, add value and create new possibilities and opportunities for existing and new organizations to explore and get useful info out of patents.

In this thesis the use of deep learning-based NLP approaches for automated patent classification and routing of patents is proposed. To achieve this, a dataset that consists of around 70,000 Greek patent applications was created. Furthermore, an English dataset was introduced for comparing and possibly augmentation purposes. The exploited algorithms are based on neural networks, examining several models (i.e., fully connected, convolutional and recurrent). In addition to this, many preprocessing methods were introduced which boost the results and assisted the algorithms. Moreover, transformer-based language models were fine-tuned. The best results were achieved by a fine-tuned version of Greek BERT, obtaining a 91,15% accuracy on three classes task and 82% accuracy on the eight classes taxonomy. Moreover, we saw similar results when comparing to the English dataset. Lastly, we examined the possibility of upscaling the existing Greek dataset with the help of using machine translation to convert them to English.

To the best of our knowledge this is the first Greek patent automated classification approach, thus, there are a lot of future steps to be done. First of all, the proposed approach is very high-level, thus, we will examine the presented models on further levels of classification. Moreover, further experimentation of lemmatization approaches is necessary to increase the accuracy of automated classification processes. The need of more data is crucial in this approach, since, collecting more well-annotated data could boost the performance. Furthermore, a dataset of a paragraph-to-paragraph similarity could assist semantic textual similarity, which is a task extremely useful for prior art search and could be combined with that of classification to develop an accurate document retrieval solution. A proof of concept of semantic similarity can be seen in Appendix C which returned good results.

Bibliography – References – Online sources

[1] A. M. Turing, Computing machinery and intelligence, Mind, vol. 59, no. October, pp. 43360, 1950

[2] A. Nucci, "Unsupervised and supervised NLP approach," Aisera, 11-May-2022. [Online]. Available: https://aisera.com/blog/unsupervised-and-supervised-nlp-approach/. [Accessed: 02-Dec-2022].

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," arXiv.org, 06-Dec-2017. [Online]. Available: https://arxiv.org/abs/1706.03762. [Accessed: 03-Jan-2023].

[4] A. Victor, "Top 5 NLP applications of Reinforcement Learning," Custom Software Engineering Services, 11-Mar-2022. [Online]. Available: https://insights.daffodilsw.com/blog/top-5-nlp-applications-of-reinforcement-learning. [Accessed: 11-Dec-2022].

[5] C. J. Fall, A. Torcsvari, K. Benzineb, and G. Karetka, "Automated Categorization in the International Patent Classification," in ACM SIGIR Forum, vol. 37, Issue 1, pp. 10-25, 2003.

[6] C. Olah, "Understanding LSTM networks," Understanding LSTM Networks -- colah's blog, 17-Aug-2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 13-Dec-2022].

[7] C. T. B. S. H. MIAP, "Recurrent neural networks and natural language processing.," Medium, 19-Apr-2021. [Online]. Available: https://towardsdatascience.com/recurrent-neural-networks-and-natural-language-processing-73af640c2aa1. [Accessed: 13-Dec-2022].

[8] D. Johnson, "What is Artificial Intelligence? introduction, history &amp; types of ai," Guru99, 19-Nov-2022. [Online]. Available: https://www.guru99.com/artificial-intelligence-tutorial.html. [Accessed: 01-Dec-2022].

[9] E. H. and J. Silge, "Supervised machine learning for text analysis in R," Chapter 8 Dense neural networks, 11-May-2022. [Online]. Available: https://smltar.com/dldnn.html. [Accessed: 12-Dec-2022].

[10] Elvis, "Deep learning for NLP: An overview of recent trends," Medium, 18-Apr-2020. [Online]. Available: https://medium.com/dair-ai/deep-learning-for-nlp-an-overview-of-recent-trends-d0d8f40a776d. [Accessed: 11-Dec-2022].

[11] H. Zellig, "Distributional structure." Word 10.2-3, pp.146-162, 1954. [11]

[12] I. Christopher and S. Sydney, and S. Spieckermann, "Automated Patent Classification," Stanford University, CS229: Machine Learning, 2011.

[13] IBM Cloud Education, "What is deep learning?" IBM, 01-May-2020. [Online]. Available: https://www.ibm.com/cloud/learn/deep-learning. [Accessed: 11-Dec-2022].

[14] J. Brownlee, "What is semi-supervised learning," MachineLearningMastery.com, 16-Dec-2020. [Online]. Available: https://machinelearningmastery.com/what-is-semi-supervised-learning/. [Accessed: 02-Dec-2022].

[15] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 4171-4186, ACL, 2018.

[16] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, vol. 1, pp. 328-339, ACL, 2018.

[17] J. Koutsikakis, I. Chalkidis, P. Malakasiotis, and I. Androutsopoulos, "GREEK-BERT: The Greeks visiting Sesame Street," 11th Hellenic Conference on Artificial Intelligence, 2020.

[18] J. Lee and J. Hsiang, "Patent Classification by Fine-Tuning BERT Language Model," in World Patent Information, vol. 61, pp. 10196

[19] J. Yun and Y. Geum, "Automated classification of patents: A topic modeling approach," in Computers & Industrial Engineering, Vol. 147, Elsevier Ltf, 2020.

[20] K. D. Foote, “A brief history of deep learning,” DATAVERSITY, 18-Mar-2022. [Online]. Available: https://www.dataversity.net/brief-history-deep-learning/#. [Accessed: 11-Dec-2022].

[21] L. Abdelgawad, P. Kluegl, E. Genc, S. Falkner, and F. Hutter, "Optimizing Neural Networks for Patent Classification," in ECML PKDD, pp. 688–703, Springer, 2019.

[22] L. Fang, L. Zhang, H. Wu, T. Xu, D. Zhou, E. Chen, "Patent2Vec: Multi-view representation learning on patent-graphs for patent classification." World Wide Web, pp. 1-22, June 2021.

[23] L. Shabao, H. Jie, C. Yuxin, and J. Jianjun, "DeepPatent: patent classification with convolutional neural networks and word embedding," in Scientometrics, Vol. 117, Springer, pp. 721-744, 2018.

[24] M. F. Grawe, C. A. Martins and A. G. Bonfante, "Automated Patent Classification Using Word Embedding," 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), 2017, pp. 408-411, doi: 10.1109/ICMLA.2017.0-127.

[25] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," in Transactions of the Association for Computational Linguistics, vol. 5, pp. 135-146, 2017.

[26] Risch, J., Garda, S., & Krestel, R. (2020). Hierarchical Document Classification as a Sequence Generation Task. In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020 (pp. 147-155).

[27] S. Cristina, “The attention mechanism from scratch,” MachineLearningMastery.com, 15-Nov-2022. [Online]. Available: https://machinelearningmastery.com/the-attention-mechanism-from-scratch/. [Accessed: 13-Dec-2022].

[28] S. Falkner, A. Klein, and F. Hutter, "BOHB: Robust and Efficient Hyperparameter Optimization at Scale," in International Conference on Machine Learning, PMLR, 2018.

[29] S. Mustafa and A. Alrifai, "Deep Learning Services for Patents," in Proceedings of the 1st Workshop on Patent Text Mining and Semantic Technologies, Karlsruhe, 2019.

[30] Scikit-learn “3.1. cross-validation: Evaluating estimator performance,” scikit. [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html. [Accessed: 30-Jan-2023].

[31] SITNFlash, “The history of Artificial Intelligence,” Science in the News, 23-Apr-2020. [Online]. Available: https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/. [Accessed: 01-Dec-2022].

[32] T. Binhuraib, “NLP with cnns,” Medium, 16-Oct-2020. [Online]. Available: https://towardsdatascience.com/nlp-with-cnns-a6aa743bdc1e. [Accessed: 13-Dec-2022].

[33] T. Kanstrén, “A look at precision, recall, and F1-score,” Medium, 19-May-2021. [Online]. Available: https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec#:~:text=F1%2DScore%20is%20a%20measure,than%20the%20traditional%20arithmetic%20mean. [Accessed: 20-Aug-2023]

[34] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," arXiv preprint arXiv:1310.4546, 2013.

[35] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[36] V. Nair and G.E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in Proceedings of the 27th International Conference on Machine Learning, Haifa, pp. 807-814, 2010.

[37] WIPO Homepage, [Online]. Available: https://www.wipo.int/portal/en/index.html, last accessed 2021/08/08

[38] WIPO, "WIPO," June 2013. [Online]. Available: https://www.wipo.int/export/sites/www/standards/en/pdf/03-09-01.pdf. [Accessed 23 OCTOBER 2022].

[39] World Intellectual Property Organization (WIPO). (2021). Patent Cooperation Treaty (PCT). [Online]. Available: https://www.wipo.int/pct/en/texts/rules/rtoc1.html [Accessed: 20-Sep-2023]

[40] X. Bing, L. Baoan, and L. Xueqiang, "Research on Patent Document Classification Based on Deep Learning," in International Conference on Artificial Intelligence and Industrial Engineering, Atlantis Press, 2016.

[41] X. Jiang, Y. Liang, Y. Chen, and N. Duan, "XLM-K: Improving Cross-Lingual Language Model Pre-Training with Multilingual Knowledge," arXiv preprint arXiv:2109.12573, 2021.

[42] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," arXiv preprint arXiv:1907.11692, 2019.

[43] Y. Verma, “A complete understanding of dense layers in neural networks,” Analytics India Magazine, 23-Apr-2022. [Online]. Available: https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/. [Accessed: 12-Dec-2022].

[44] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations," arXiv preprint arXiv:1909.11942v1, 2020.

Appendix A

Weight and Bias

Hyperparameter tuning refers to the process of adjusting the values of the weights and biases in a neural network to optimize its performance. This is typically done through techniques such as grid search or random search, where different combinations of weight and bias values are tested and evaluated using a metric such as accuracy or loss. The goal is to find the set of values that results in the best performance on the training or validation data.

Weights and Biases (W&B) is a platform that provides tools for machine learning experimentation and model versioning. It allows users to track, compare and collaborate on their machine learning experiments through the use of a simple python API.

W&B features include:

- Automated logging of model weights, gradients, inputs, outputs, and other metrics,
- Visualization of performance metrics, gradients, and other data,
- Comparison of different models and their performance,
- Collaboration with team members,
- Automated model versioning and rollback,
- Integration with popular machine learning frameworks such as TensorFlow, PyTorch, and Keras.

In summary, W&B is a platform that helps data scientist to keep track of their machine learning experiments and easily compare and collaborate on them.

In this research, the LSTM model were chosen to be hyper tuned. This is because, as a model, it is the most promising for this task. This limitation was because this procedure is very time consuming and resource consuming. The best parameters that were found after a grid search as seen in Figure A.1 and Figure A.2, were:

- Best size for LSTM layers: 32
- Best size for 1st FC layer: 512
- Best size for 2nd FC layer: 256
- Best dropout: 0.3
- Best Learning Rate: 0.001
- Best Batch Size: 64

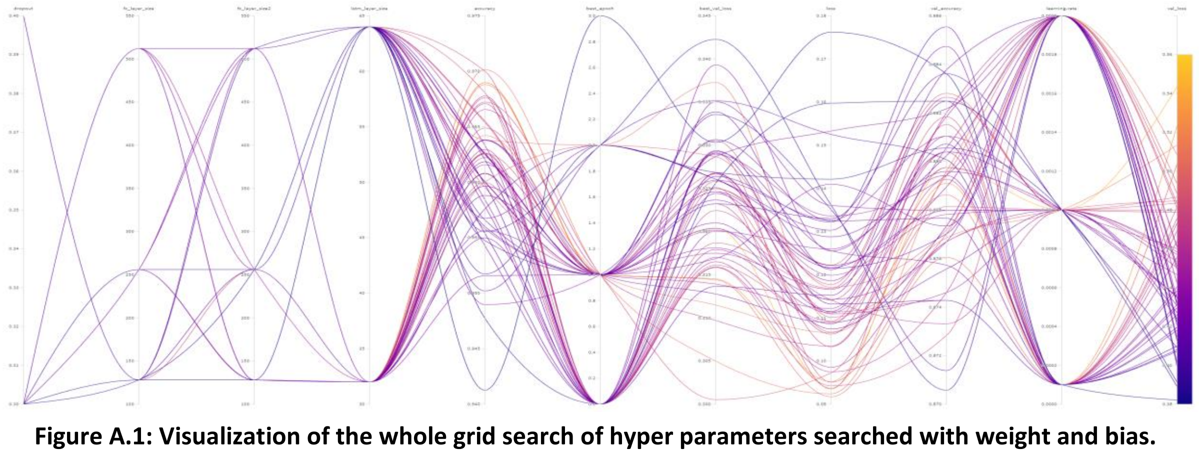


Figure A.1: Visualization of the whole grid search of hyper parameters searched with weight and bias.

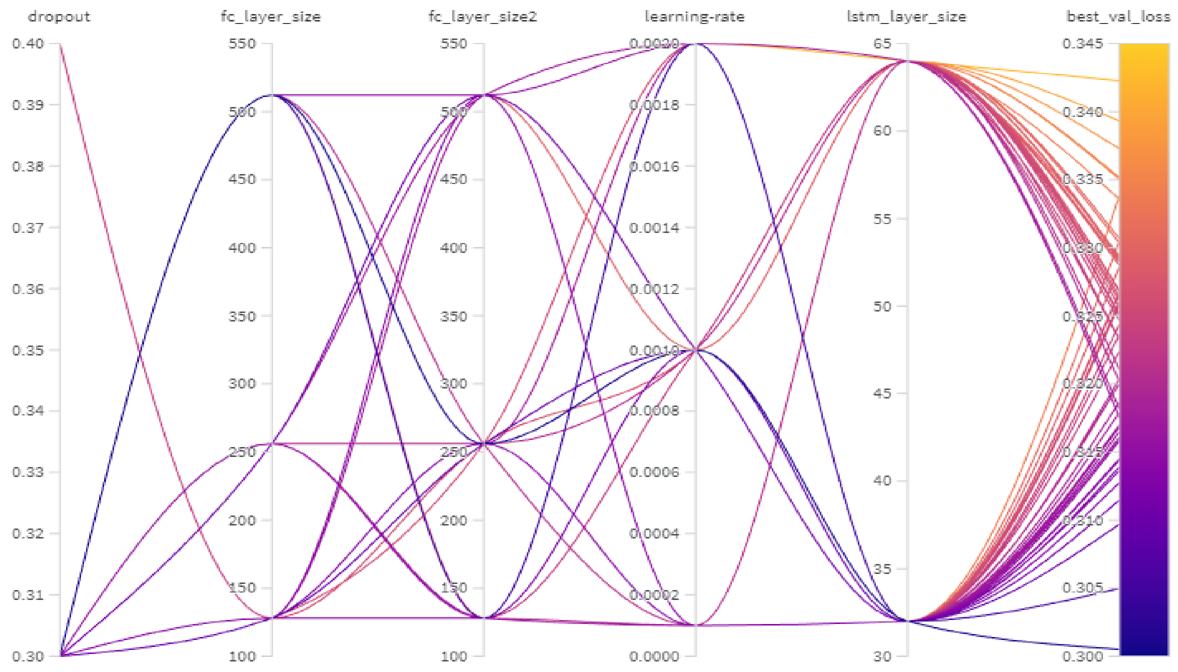
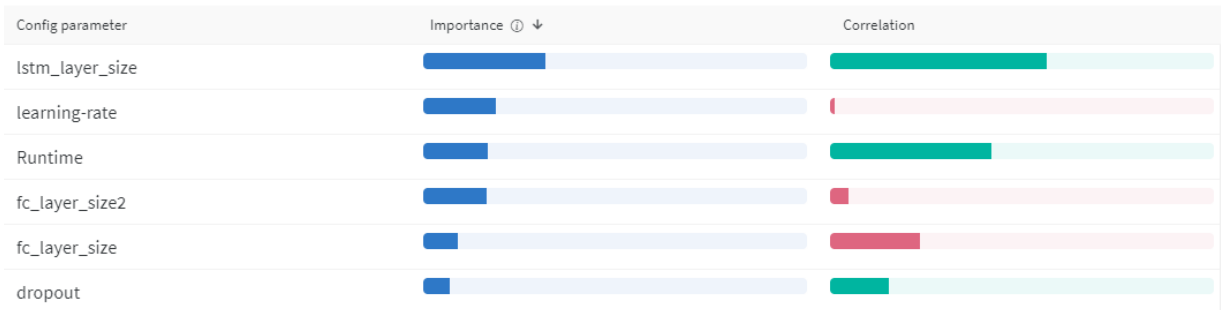


Figure A.2: Visualization of the most important grid search of hyper parameters searched with weight and bias.

For a better view, the most important having more impact on the model were the size of the layers and the learning rate. This can be seen on how much is the impact in Figure A.3.



Config parameter	Importance ⓘ ↓	Correlation
lstm_layer_size		
learning-rate		
Runtime		
fc_layer_size2		
fc_layer_size		
dropout		

Figure A.3: Parameter-Importance-Correlation graph after hyper tuning using Weight and Bias tool.

Results after validating the best parameters of what grid search via Weight and Bias tool indicated was 89% validation accuracy and better results from all the other trials before that as can been seen in Table A.0.1.

Table A.1. Classification Report for the LSTM in the 3 Classes problem after hyper tune.

	Precision	Recall	F1-score	Support
ΗΛΕΚΤΡΟΛΟΓΙΚΟ	0.79	0.74	0.77	892
ΜΗΧΑΝΟΛΟΓΙΚΟ	0.81	0.90	0.86	2199
ΧΗΜΙΚΟ	0.96	0.91	0.93	3811
ACCURACY	-	-	0.89	6902
MACRO AVG	0.85	0.85	0.85	6902
WEIGHTED AVG	0.89	0.89	0.89	6902

Appendix B

Dataset Class Weights

GPD-3

Class weights for 3 classes problem:

Mechanical:	2.6621923937360177
Electrical:	1.0421573956634655
Chemical:	0.6006648913024559

GPD-8

Class weights for 8 classes problem:

A:	0.3667375132837407
B:	0.9090190706985565
C:	0.44503765604044154
D:	14.524410774410775
E:	2.6611659469463294
F:	1.9742562929061784
G:	2.3311267225074306
H:	1.8313521545319464

Appendix C

Another approach was to get a semantic similarity of each patent abstract. The goal was to get a semantic similarity of a whole paragraph and be able to compare it with other paragraphs. This was done by creating a representation of the whole paragraph with the use of the embeddings each paragraph contained.

The method that was used, was to take the average of all embeddings of all the words that a paragraph had and create a new space with embeddings of paragraphs instead of embeddings of words. By getting the cosine similarity, it was able to return the most similar paragraph to the one that we were targeting, thus the most similar in distance was the most similar semantically.

Searched:

"*Η παρούσα εφεύρεση αφορά σε χημικές ενώσεις με γενικό τύπο (I) στον οποίο τα R1 και R2, τα οποία μπορεί να είναι τα ίδια ή διαφορετικά, συμβολίζουν αλκύλιο με 1 έως 4 άτομα άνθρακα, και το R3 συμβολίζει υδρογόνο, αλογόνο, αλκύλιο με 1 έως 4 άτομα άνθρακα, ή O-αλκύλιο με 1 έως 4 άτομα άνθρακα, και στους δυνάμενους να υδρολυθούν σε εσωσωματική δοκιμασία εστέρες αυτών με φαρμακευτικά αποδεκτά οξέα. Οι παρούσες χημικές ενώσεις έχουν αξία στην εξάσκηση του ιατρικού επαγγέλματος προς όφελος των ανθρώπων ή των ζώων.*"

Most similar document based on cosine similarity:

“*Η εφεύρεση αφορά σε ενώσεις με χημικό τύπο I, ή σε στερεοϊσομερή, σε φαρμακευτικώς αποδεκτά άλατα αυτών ή σε φαρμακευτικώς αποδεκτά άλατα των προφαρμάκων αυτών. Η εφεύρεση αυτή αφορά επίσης σε φαρμακευτικές συνθέσεις που περιέχουν μία ένωση με χημικό τύπο I, και σε μεθόδους για την θεραπεία του διαβήτη, της αντοχής στην ινσουλίνη, της διαβητικής νευροπάθειας, της διαβητικής νεφροπάθειας, της διαβητικής αμφιβληστροειδοπάθειας του καταρράκτη, της υπεργλυκαιμίας, της υπερχοληστερολαιμίας, της υπέρτασης, της υπερινσουλιναιμίας, της υπερλιπιδαιμίας, της αθηροσκλήρωσης, ή της ισχαιμίας των ιστών.”*

After evaluating few documents, the semantic similarity using cosine similarity could give some good results, but it needs to be improved. Cosine similarity returns a list of similar documents having each one a similarity score but sorting them did not provide always the most similar documents first. However, it could be combined in the future with a classifier to provide more accurate results.