Solving University Entrance Assessment Using Information Retrieval

Igor Cataneo Silveira

Thesis presented to
the
Institute of Mathematics and Statistics
from
Universidade de São Paulo
to
obtain the title
of
Master in Sciences

Program: Graduation in Computer Science

Advisor: Prof. Dr. Denis Deratani Mauá

During the development of this work the author received financial support from CAPES

São Paulo, June of 2018

Solving University Entrance Assessment Using Information Retrieval

This thesis version has the corrections and alterations proposed by the Evaluation Committee during the defense of the original work, held in 2018. A copy of the original work is available at Institute of Mathematics and Statistics of the Universidade de São Paulo.

Evaluation Committee:

- Prof. Dr. Denis Deratani Mauá (advisor) - IME-USP
- Prof. Dr. Fabio Gagliardi Cozman - POLI-USP
- Prof. Dr. Diana Maria de Sousa Marques Pinto dos Santos - University of Oslo

Acknowledgements

I would like to thank Professor Denis Deratani Mauá not only because he accepted me as his student, but also for the caring he showed in my first months in São Paulo, for his advises about my possible career, for he did not just correct my texts, but also had the patience to try to improve my writing. I will be very proud of myself if someday my scientific writing becomes half as good as his.

Unfortunately I cannot describe the importance everyone had during these two years, but I would like to "carve in stone" the names of everyone that was present physically or virtually during this time: Aline "Fantástica" Carolina Oliveira, **Amanda Angelo Guerra**, Amanda Rabelo Palma, Ana Melisa Paiba Amaya, Anderson "Handick" Luiz Ferreira de Menezes, Ana Paula Stienen, Augusto Rotta Filho, Carlos "Lee" Celso de Moura Junior, Carolina Aline Palma, Christopher Emannuel Salgado Guimarães, Cristiane Jéssica Babinski, Daniela "Mogu" Businari Pollesi, Fábio Mussoi, Flávio "Lesh" Murakami, Francielli Vilela Peres, Gabriel Alves Godoy, Gabriel Reis, Gabriela Veiga Dias, Gabriella Dias de Assunção, Gabriella da Rocha Soletti, George "Rapazola" Augusto Moreira Czelusniak, Giulia Alberti, Guilherme Brittes Benitez, Guilherme Vargas, Guilherme "Piá" de Castro, Gustavo Veloso Tomio, Gustavo "Guga" Vitor Barbosa Bomfim, Igor "RM10" Luis Corradi, Igor Thales Reginaldo Neves, Isac — O Gato, Jonas Vieira de Assis, Jordana "Jojo" Sarmenghi Salamon, João "Keda" da Silva, Karina Suemi Awoki, Krupskaya Kassandra Pacheco Carhuarupay, Laís "Helena" Muntini, Laryssa Akemi Murakami, **Letícia Belão Gumiero da Silva**, Lin Chi Yu, Lin Yu Han, Lorena Perdigão Nocera, Luan Willian Pinto Bueno Dias, **Luciano Walenty Xavier Cejnog**, Luiz Henrique "KV" Bernardon, Magno Marcos Miotto Parmigiani, Mateus "Betoneira" Bruschi, Murilo Caio Mayer, Newton Schner Junior, Pablo Kyoshi Rocha, Pablo Sandino Ferreira Botelho, Pedro Bertolli, Pedro de Oliveira Vianna, Pedro Otávio Zolini Ortelani, Rafael Lourenço, **Rafael Veiga Pocai**, Rafael Vendrusculo, Raúl Leonardo Rincon, Renan "Bong" de Freitas Fantinelli, Renan Ramon "Dalzoto" Ramos Mendes, Rodrigo Lourenço, Sergio Francisco das Chagas Júnior, Stephany Cestari Guimarães, Willian "Wii" Abreu Ferreira, Yan Rodrigues do Prado Chapine and Zoneibe Augusto Silva Luz.

At last but not least, I would like to thank Sandra Aparecida de Assunção Silva, for daily providing two to four sacred cups of coffee.

[I swear that the next one will be to my family.]

Resumo

SILVEIRA, I. C. **Solving University Entrance Assessment Using Information Retrieval**. 2018. 75f. Dissertação - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2018.

Responder perguntas feitas em linguagem natural é uma capacidade há muito desejada pela Inteligência Artificial. Porém, produzir um sistema de Question Answering (QA) é uma tarefa desafiadora, uma vez que ela requer entendimento de texto, recuperação de informação, extração de informação e produção de texto. Além disso, a tarefa se torna ainda mais difícil dada a dificuldade em coletar datasets confiáveis e em avaliar as técnicas utilizadas, sendo estes pontos de suma importância para abordagens baseadas em aprendizado de máquina. Isto tem levado muitos pesquisadores a focar em Multiple-Choice Question Answering (MCQA), um caso especial de QA no qual os sistemas devem escolher a resposta correta dentro de um grupo de possíveis respostas. Um caso particularmente interessante de MCQA é o de resolver testes padronizados, tal como testes de proficiência linguística, teste de ciências para ensino fundamental e vestibulares. Estes exames fornecem perguntas de múltipla escolha de fácil avaliação sobre diferentes domínios e de diferentes dificuldades.

O Exame Nacional do Ensino Médio (ENEM) é um exame realizado anualmente por estudantes de todo Brasil. Ele é utilizado amplamente por universidades brasileiras como vestibular e é o segundo maior vestibular do mundo em número de candidatos inscritos. Este exame consiste em escrever uma redação e resolver uma parte de múltipla escolha sobre questões de: Ciências Humanas, Linguagens, Matemática e Ciências Naturais. As questões nestes tópicos não são divididas por matérias escolares (Geografia, Biologia, etc.) e normalmente requerem raciocínio interdisciplinar. Ademais, edições passadas do exame e suas soluções estão disponíveis online, tornando-o um benchmark adequado para MCQA.

Neste trabalho nós automatizamos a resolução do ENEM focando, por simplicidade, em questões puramente textuais que não requerem raciocínio matemático. Nós formulamos o problema de responder perguntas de múltipla escolha como um problema de identificar a alternativa mais similar à pergunta. Nós investigamos duas abordagens para medir a similaridade textual entre pergunta e alternativa. A primeira abordagem trata a tarefa como um problema de Recuperação de Informação Textual (IR), isto é, como um problema de identificar em uma base de dados qualquer qual é o documento mais relevante dado uma consulta. Nossas consultas são feitas utilizando a pergunta mais alternativa e utilizamos três diferentes conjuntos de texto como base de dados: o primeiro é um conjunto de artigos em texto simples extraídos da Wikipedia em português; o segundo contém apenas o texto dado no cabeçalho da pergunta e o terceiro é composto por pares de questão-alternativa correta extraídos de provas do ENEM. A segunda abordagem é baseada em Word Embedding (WE), um método para aprender representações vetoriais de palavras de tal modo que palavras semanti-

camente próximas possuam vetores próximos. WE é usado de dois modos: para aumentar o texto das consultas de IR e para criar representações vetoriais para a pergunta e alternativas. Usando essas representações vetoriais nós respondemos questões diretamente, selecionando a alternativa que maximiza a semelhança de cosseno em relação à pergunta, ou indiretamente, extraindo features das representações e dando como entrada para um classificador que decidirá qual alternativa é a correta. Junto com as duas abordagens nós investigamos como melhorá-las utilizando a WordNet, uma base estruturada de dados lexicais onde palavras são conectadas de acordo com algumas relações, tais como sinonímia e hiperonímia. Por fim, combinamos diferentes configurações das duas abordagens e suas variações usando WordNet através da criação de um comitê de resolvedores encontrado através de uma busca gulosa. O comitê escolhe uma alternativa através de voto majoritário de seus constituintes.

A primeira abordagem teve 24% de acurácia utilizando o cabeçalho, 25% usando a base de dados de pares e 26.9% usando Wikipedia. A segunda abordagem conseguiu 26.6% de acurácia usando WE indiretamente e 28% diretamente. O comitê conseguiu 29.3%. Estes resultados, pouco acima do aleatório (20%), sugerem que essas técnicas conseguem captar algumas das habilidades necessárias para resolver testes padronizados. Entretanto, técnicas mais sofisticadas, capazes de entender texto e de executar raciocínio de senso comum talvez sejam necessárias para alcançar uma performance humana.

Palavras-chave: Multiple-Choice Question Answering, ENEM, Recuperação de Informação.

Abstract

SILVEIRA, I. C. **Solving University Entrance Assessment Using Information Retrieval**. 2018. 75f. Thesis - Institute of Mathematics and Statistics, Universidade de São Paulo, São Paulo, 2018.

Answering questions posed in natural language is a key task in Artificial Intelligence. However, producing a successful Question Answering (QA) system is challenging, since it requires text understanding, information retrieval, information extraction and text production. This task is made even harder by the difficulties in collecting reliable datasets and in evaluating techniques, two pivotal points for machine learning approaches. This has led many researchers to focus on Multiple-Choice Question Answering (MCQA), a special case of QA where systems must select the correct answers from a small set of alternatives. One particularly interesting type of MCQA is solving Standardized Tests, such as Foreign Language Proficiency exams, Elementary School Science exams and University Entrance exams. These exams provide easy-to-evaluate challenging multiple-choice questions of varying difficulties about large, but limited, domains.

The Exame Nacional do Ensino Médio (ENEM) is a High School level exam taken every year by students all over Brazil. It is widely used by Brazilian universities as an entrance exam and is the world's second biggest university entrance examination in number of registered candidates. This exam consists in writing an essay and solving a multiple-choice test comprising questions on four major topics: Humanities, Language, Science and Mathematics. Questions inside each major topic are not segmented by standard scholar disciplines (e.g. Geography, Biology, etc.) and often require interdisciplinary reasoning. Moreover, the previous editions of the exam and their solutions are freely available online, making it a suitable benchmark for MCQA.

In this work we automate solving the ENEM focusing, for simplicity, on purely textual questions that do not require mathematical thinking. We formulate the problem of answering multiple-choice questions as finding the candidate-answer most similar to the statement. We investigate two approaches for measuring textual similarity of candidate-answer and statement. The first approach addresses this as a Text Information Retrieval (IR) problem, that is, as a problem of finding in a database the most relevant document to a query. Our queries are made of statement plus candidate-answer and we use three different corpora as database: the first comprises plain-text articles extracted from a dump of the Wikipedia in Portuguese language; the second contains only the text given in the question's header and the third is composed by pairs of question and correct answer extracted from ENEM assessments. The second approach is based on Word Embedding (WE), a method to learn vectorial representation of words in a way such that semantically similar words have close vectors. WE is used in two manners: to augment IR's queries by adding related words to those on the query according to the WE model, and to create vectorial representations for statement and candidate-answers. Using these vectorial representations we answer questions either directly, by

selecting the candidate-answer that maximizes the cosine similarity to the statement, or indirectly, by extracting features from the representations and then feeding them into a classifier that decides which alternative is the answer. Along with the two mentioned approaches we investigate how to enhance them using WordNet, a structured lexical database where words are connected according to some relations like synonymy and hypernymy. Finally, we combine different configurations of the two approaches and their WordNet variations by creating an ensemble of algorithms found by a greedy search. This ensemble chooses an answer by the majority voting of its components.

The first approach achieved an average of 24% accuracy using the headers, 25% using the pairs database and 26.9% using Wikipedia. The second approach achieved 26.6% using WE indirectly and 28% directly. The ensemble achieved 29.3% accuracy. These results, slightly above random guessing (20%), suggest that these techniques can capture some of the necessary skills to solve standardized tests. However, more sophisticated techniques that perform text understanding and common sense reasoning might be required to achieve human-level performance.

Keywords: Multiple-Choice Question Answering, ENEM, Information Retrieval.

Contents

Abbreviations List

AH	Adding Heuristic
AI	Artificial Intelligence
CBOW	Continuous-Bag-Of-Words
CE	Chemical Elements
DS	Domain Specific
EK	Encyclopedic Knowledge
ENEM	Exame Nacional do Ensino Médio
FOL	First Order Logic
GloVe	Global Vectors
IC	Image Comprehension
ILP	Integer Linear Programming
IR	Information Retrieval
KB	Knowledge Base
MCQA	Multiple-Choice Question Answering
MLN	Markov Logic Networks
MR	Mathematical Reasoning
MRR	Mean Reciprocal Rank
NDH	Non-Deciding Heuristic
POS	Part-of-Speech
QA	Question Answering
SQUABU	Science Questions Appraising Basic Understanding
SVM	Support Vector Machine
TC	Text Comprehension
WE	Word Embedding
W2V	Word2Vec

List of Figures

List of Tables

Chapter 1

Introduction

Automatically answering questions posed in natural language is a long desired goal of Artificial Intelligence. This task, in its most generic form, consists of answering a question posed in free-text format by querying a knowledge base (KB), identifying relevant information and producing a final answer in natural language. Besides the intrinsic difficulties of these sub-tasks, there are two major hindrances that challenge effectively designing Question Answering (QA) systems. First, solutions are restricted by their KB. Curated and structured KB are costly to build; consequently they are usually limited to narrow domains, while questions usually encompass much larger domains. Second, given an output answer, it is not trivial to determine whether the output is a valid answer or how good it is.

In order to mitigate these issues, Miyao and Kawazoe proposed using university entrance exams as a less ambitious, but still challenging, benchmark for QA [MK13]. In fact, solving standardized tests, such as university entrance exams, is a type of Multiple-Choice QA (MCQA), a sub-task that dispenses with the need of producing sentences in natural language and whose answers can be automatically and objectively evaluated. The proposed exam comprises 11 subjects: Japanese, English, Mathematics, World History, Japanese History, Modern Society, Politics & Economics, Ethics, Physics, Chemistry and Biology. All of the subjects but English are written in Japanese, and all the questions, except the ones on Mathematics, are multiple-choice questions. The authors created a collection of MCQA problems by manually translating the exam's questions into a machine-readable format. In a 2014 competition, contestant systems were unable to match overall human performance; the best performance achieved was 58% in World History and the worst performance achieved was 26% in English [FKKM14]. As a matter of fact, this dataset can be seen as eleven topic-specific datasets. This subject separation allows for using AI techniques to explore specific characteristics of a subject in this exam, thus not generalizing properly neither to other subjects nor to other languages. Furthermore, this dataset is not publicly available.

A similar proposal was put forward by Cheng et al. [CZW+16], who addressed the problem of solving the *Gaokao*, a Chinese university entrance exam. This exam has three mandatory subjects — Chinese, Mathematics and Foreign Language — and three other subjects from a group that the test-taker must choose: either Geography, History and Politics or Physics, Chemistry and Biology. The authors translated the questions into a machine-readable format and made available only the questions from the History subject, in which they report achieving 44% accuracy. As with the Japanese benchmark, the segmentation into known topics makes this dataset topic-specific and with a relatively narrow domain.

The Exame Nacional do Ensino Médio (ENEM) is an advanced High School level exam applied every year all over the country by the Brazilian government. In 2016 9.2 million people registered for the exam, falling shortly after the Gaokao exam, which had 9.4 million registered students [Bra16, Yua16]. The ENEM is used by many Brazilian universities as an entrance exam, and recently became part of an unified admission procedure followed by all Federal Public Universities in Brazil. The ENEM assessment is divided in writing an essay and a multiple-choice exam comprising four major areas: Humanities, Language, Science and Mathematics. The questions are not segmented into

subjects and many questions combine more than one scientific discipline. Importantly, exams and their solutions are publicly and freely available at a govern-hosted website. In this work we propose using the ENEM as a new benchmark for (MC)QA. In comparison with the previous university entrance exams used for (MC)QA, the ENEM has the following advantages: it is available, it is bigger — in number of available questions — than the others, it is not topic-specific and promotes Natural Language Processing for Brazilian Portuguese.

1.1 Goals

The goals of this work are to create a machine-readable dataset of questions from the ENEM exam and to develop baseline methods that will foster research on the topic. For simplicity we focus on purely textual questions that do not rely on non-textual understanding. That is, we do not use questions referring to diagrams, chemical or mathematical formulae, charts, drawings and pictures in general.

1.2 Contributions

We created an annotated machine-readable dataset of multiple-choice textual questions taken from the ENEM exams between 2009 and 2017. In this dataset the questions are annotated with structural tags, segmenting the question into header, statement and alternatives, and informative tags to help in the performance analysis of the techniques. Informative tags flag if the question uses images, chemical formulas and what type of knowledge the question requires.

We evaluated the performance of several similarity-based techniques in answering questions from our dataset. In order to measure text similarity we investigate two different approaches: Information Retrieval and Word Embedding. Additionally we consider ways of enhancing the previous approaches bringing knowledge from WordNet. Moreover we present a way to combine these approaches: a greedy search and majority voting. Finally, we used Word Embedding to extract features from text and used them as input to a Support Vector Machine.

Information Retrieval (IR) is a technique based on identifying which documents from a database are most relevant to obtain a piece of information [RN02]. This is usually done by retrieving and scoring documents by their similarity to a query document, where similarity can be taken as number of matched words, put differently, words that occur in both the query and the retrieved document. This approach is used to answer multiple-choice questions by building a knowledge base containing text documents, then searching for common words of statement plus answer with those on the documents of the knowledge base. We used three different bases: one is made of the header given in the question, one is composed by pairs of question-answer taken from the ENEM and the last is composed by articles extracted from the (Brazilian) Portuguese Wikipedia. Also, we combine these three sources in two alternative ways: summing their scores or using them sequentially. The best scoring algorithm of this family achieved 26.9% accuracy, that is, slightly above random guessing (20%).

The Word Embedding (WE) method is a neural network based method to learn vector representations for words such that semantically similar words are represented by near vectors [MSC+13]. Consequently, it can be used to verify similarity between phrases, being useful to answer questions which the answer is the most similar alternative to the statement. Additionally, this approach can be used to augment phrases to enhance IR's search. Another way of using Word Embedding to answer questions is using them indirectly, as proposed by [CEK+16]. That is, by extracting features from answer-statement pairs and using them as inputs to a SVM classifier; the goal is to use these features in order to classify the pair as a true alternative or a false alternative. This approach achieved 28.1% accuracy when selecting the most similar candidate-answer, while the features-based classifier scored 26.6%.

The lexical ontology WordNet is used to augment the text of the questions, thus enhancing the performance of the two previous approaches. This ontology is used to insert into a phrase the

synonyms of the words occurring in it, or, alternatively we can use not the synonyms, but the hypernyms, hyponyms or holonyms. Using this tool we introduce a kind of "external knowledge" to the algorithms.

Finally, we unite the two approaches and their WordNet variations using an ensemble of algorithms. We expect that algorithms in the ensemble capture different aspects of a question and by combining answers they can achieve a better performance together than individually, like Allen's Aristo and IBM's Watson [CEK+16, FBCC+10]. This ensemble is created through a greedy search of combinations. The ensemble decides the answer by the majority voting of its components. This final form achieved 29%, improving over the algorithms individually, suggesting that they capture some aspects required to solve the questions. However, the final result is not far from the random guesser, indicating that more sophisticated techniques capable of, for instance, commonsense reasoning and text understanding are required to match human performance.

1.3 Organization of the Chapters

The rest of the document is organized as follows. First we present in Chapter 2 the definition of Information Retrieval and Word Embedding. Following, in Chapter 3 is our literature review: similar tests that are being solved in English (Fourth Grade Science Test), Japanese (Center Test) and Chinese (Gaokao). Then, in Chapter 4, we describe the creation of the ENEM dataset: the exam, the structure of the XML version and the informative tags that we added. Next, Chapter 5 has the explanation of our methods, how we used Information Retrieval, Word Embedding, SVM, WordNet and the greedy search. The performance of these techniques is presented in Chapter 6; the discussion of these results is presented in Chapter 7, where we review the results, compare the performance of the (best) algorithms per knowledge tag and show the questions that every individual algorithms got wrong. In Chapter 8 we present some related works: characteristics of QA systems, chatbots, some QA tests that were proposed as replacements for the Turing Test and a Portuguese initiative of language processing. Finally, we conclude in Chapter 9 by reviewing the proposal, results and future work.

Chapter 2

Foundations

In this chapter we present the relevant background on Information Retrieval and Word Embedding, two key concepts through this work.

2.1 Information Retrieval

Information Retrieval (IR) is a technique based on identifying relevant documents to obtain a piece of information [RN02, MRS08]. While this definition is wide-ranging enough to be applied for texts, images, videos, etc., in this work we are interested only in texts. Bearing in mind that our documents are always text, our problem is: to identify from a collection of documents a subset of relevant documents given a query consisting of a short fragment of text. We focus here on the so-called indirect selection or ranking, which ranks documents by their relevance to the query and then returns the top ranked documents.

Consider a document space composed by documents D_i, each document is composed by index terms[1] T_j, these terms can be weights or Boolean — stating the existence (1) or not (0) of that term in that document. In this case a document D_i can be represented by a vector belonging to $\mathbb{R}^{\mathrm{T}}$:

$$D_i = [d_{i1},\ d_{i2}, ...,\ d_{1T}]$$

Where d_{ij} is the weight of the jth term of D_i and T is the size of the vocabulary. This approach permits documents to be represented as vectors in a T-dimensional space, called *vector space model*. The vocabulary is extracted directly from the document space, it is the list of (different) terms that occur in this space [SWY75]. An example of matrix Vocabulary by Documents is given in Figure 2.1. In this matrix each cell is the Boolean value of the term in that document. "Caesar", for instance, occurs in all documents except in The Tempest, and in the document Othello also occur the terms "mercy" and "worser".

Usually the weight d_{ij} is the tf-idf of the term j in the document i. More formally, to attribute a weight to a term in a document three other concepts/formulae are used: the Term Frequency (tf), the Document Frequency (df), and the Inverse Document Frequency (idf). The tf is a function over a term and a document: it is the square root of the number of times that the term occurs in that given document. That is, a term used frequently has a higher weight. Df is the number of documents that contain a given term, idf is the inverse of this value. The intuition is that a rare term should be more important. Finally, the tf-idf of a term in a document is the multiplication of its tf by its idf. So, given a term T_j and a document D_i, the weight d_{ij} equals the tf-$idf(T_j, D_i)$:

$$tf(T_j, D_i) = (count(T_j)\ in\ D_i)^{\frac{1}{2}}$$

$$idf(T_j) = 1 + log(\frac{1}{DF(T_j) + 1})$$

[1] The mapping of raw words into terms by the tokenizer is not discussed in this work. We take as input the documents already indexed.

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello
Antony	1	1	0	0	0
Brutus	1	1	0	1	0
Caesar	1	1	0	1	1
Calpurnia	0	1	0	0	0
Cleopatra	1	0	0	0	0
mercy	1	0	1	1	1
worser	1	0	1	1	1
...					

Figure 2.1: *Example of a matrix of Vocabulary by Documents from [MRS08].*

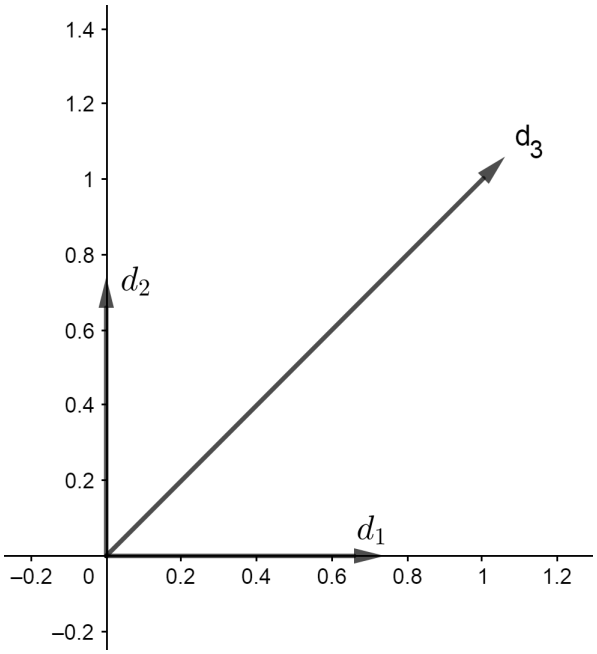


Figure 2.2: *Representation of three documents in a bi-dimensional space*

$$tf\text{-}idf(T_j, D_i) = tf(T_j, D_i) \times idf(T_j)$$

Consider the case of Vocabulary = {king, queen}, thus a bi-dimensional space, and three documents D_1, D_2 and D_3, the first containing "king, king", the second "queen, queen" and the third containing four occurrences of "king" and four of "queen". They are represented in this bi-dimensional space as showed in Figure 2.2, where $D_1 = [0.74, 0]$, $D_2 = [0, 0.74]$ and $D_3 = [1.06, 1.06]$.

The central point in scoring in this model is that a query is treated as a document of the same space, this way documents can be ranked by their cosine similarity. The cosine similarity is measured as the dot product $(\cdot)$ of two vectors over the product of their length. Being $D_q \in \mathbb{R}^T$ the document of the query and $D_i \in \mathbb{R}^T$ a document of the database, their similarity is measured by:

$$\text{cosine similarity} \ = \ \frac{D_q \cdot D_i}{|D_q||D_i|}$$

Given a query we compute its similarity with all documents and sort them by decreasing order of similarity. The final step is to fetch the first N documents of this list. The software we use in our work, Lucene, uses cosine similarity as base of its scoring formula with the following alterations: the tf of a term in a query is always one and it divides the dot product only by the query's length. Thus, Lucene's scoring function for a given query D_q and a document D_i is:

$$Score(D_q, D_i) = coord(D_q, D_i) \times queryNorm(D_q) \times \sum_{T_j \in D_q} tf(T_j, D_i) \times idf(T_j)^2 \times norm(D_i)$$

Where *coord* is the number of query's terms that occur in the document. $QueryNorm$ equals one over the length of D_q. The final term, $norm(t, d)$ is a penalizing factor for longer documents, it is equal one over the square root of the number of terms in that document.

These approaches, although useful, cannot handle two ubiquitous characteristics of natural languages, namely, the existence of polysemy and synonymy. Consider the case of a document containing a synonym of a word used in the query, the score of this document will be lower than it should be. With polysemy we have the opposite: a word used with a different meaning in a different context makes the score of the the document higher than it should be. To deal with this problem there is a field or a task called Latent Semantic Analysis or Latent Semantic Indexing which studies how to compress, say, a billion dimensions vocabulary space into a hundred dimensions space. This is performed by decomposing the Vocabulary-Document matrix into other three that, when multiplied, the result *approximates* the original matrix. This approach is called *low-rank approximation* [MRS08, Chapter 18]; it roughly corresponds to transforming the sparse representation of a term into a dense one. Even though Latent Semantic Indexing can improve the performance of IR systems, obtaining a low-rank matrix decomposition of large collection of documents is too expensive computationally and is usually not used. In the next session we discuss Word Embedding, a way to estimate low-rank approximation for terms.

2.2 Word Embedding

Word2Vec (W2V) is a neural network method for learning vector representations of words[2] such that words appearing in similar contexts are represented by near vectors [BDK14, MSC+13]. W2V have been shown capable of answering semantic and syntactic analogies [MCCD13] such as "Man is to King like Woman is to ?" and "Fly is to flying like cry is to ?". Being $vector$(x) the vector associated to x, the previous analogy can be answered by finding the closest vector to $vector$(king) − $vector$(man) + $vector$(woman), which is expected to be $vector$(queen). Other given example is that $vector$(Germany) + $vector$(capital) is close to $vector$(Berlin). In this case similarity is usually measured by the cosine similarity.

This neural network has three layers: input, hidden and output layers. The input and the output layers have size T, where each position represents a word. Each word is thus represented by having one in its representing position and zero in all the others, this is called "one hot vector". In the output layer each position contains the probability of that word occurring given the input.

W2V has two models: Skip-Gram and Continuous Bag of Words (CBOW), their goal is the same — to learn dense vectorial representation of words —, but their task is slightly different. In the Skip-Gram model the task is to predict the context of a given word, where context is a fixed size window of words occurring before and after the given word. In other words, given a target word W_t we want to determine the probability of finding another word W_2 in its context. Alternatively, CBOW model's task is to predict a word given its context. That is, the input of this model is the context and we want to determine the probability of finding W_t in that context. Figure 2.3 presents the difference of the two models, having context defined as the two words before and after the target. It is important to state that the position of the words in the context is of no importance, W2V treats all phrases as a bag of words.

The hidden layer of this process is a weighted matrix of T rows and S columns. The number of columns represents the number of dimensions where the words will be "allocated" and is usually a relatively small number — less than one thousand. The model has to adjust these weights in order

[2]What in Information Retrieval is called a term here is a word. In this section we use the terminology "word" because it is more natural to talk about Word Embedding of words instead of the Word Embedding of terms.

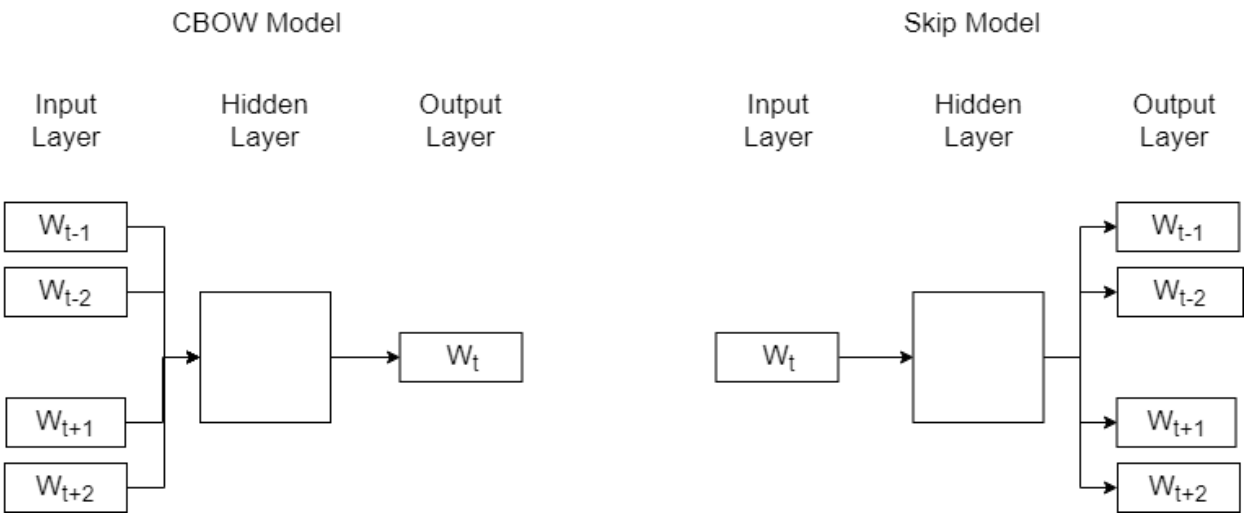


Figure 2.3: *Difference between CBOW and Skip models.*

to predict properly the probabilities. In the end of this process words that occur in similar contexts should have similar weights in this matrix, thus having close vectors.

These models have a set of hyper-parameters, such as: the size k of the context window and the vector size S. Figure 2.4 depicts examples of window contexts of size two. Larger values for k tend to create better word representations, but also makes the training more expensive.

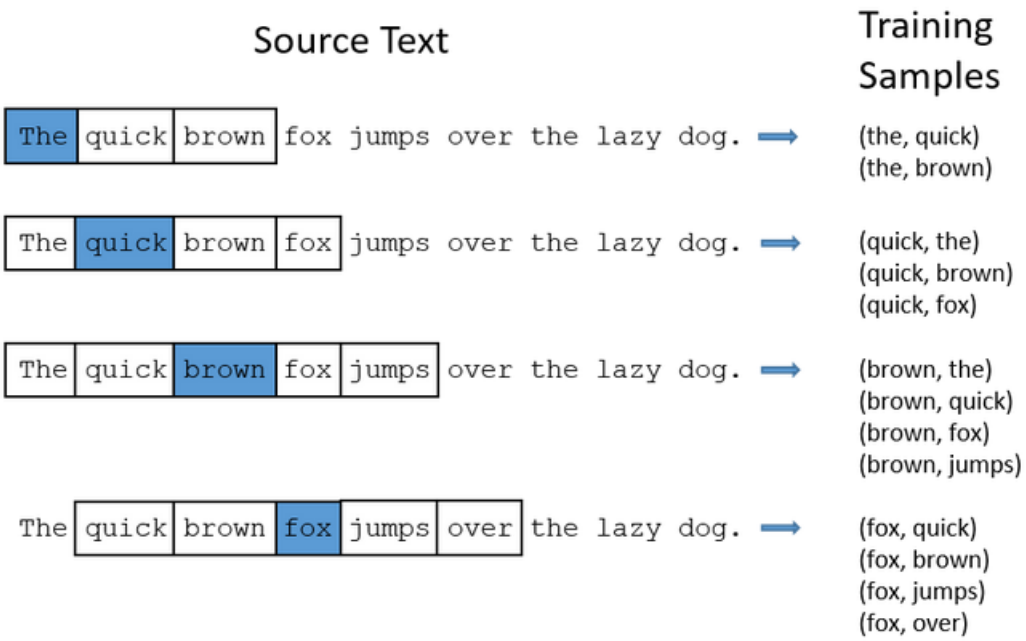


Figure 2.4: *Examples created using a window context of size 2. Taken from [McC16].*

Several extensions have been proposed recently such as FastText [BGJM17], Wang2Vec [LDBT15] and Global Vectors [PSM14].

The FastText extension proposes a model that optimizes morphological similarity between words by taking into account the subwords of each word in the vocabulary. The authors propose learning not only the vector representation for the whole word, but also for the n-grams of characters that compose the word — these parts will be shared across different words — plus a special case: the word itself. For instance, the 3-grams for "Lebensessenz" are:

$$< Le, Leb, ebe, ben, ene, nes, ess, sse, sen, enz, nz, Lebensessenz >$$

Table 2.1: *Expectations of the FastText, Wang2Vec and GloVe models and what they achieved in Brazilian Portuguese*

model	expectation	achieved in PTBR (accuracy %)	vector size
FastText	good in syntactic analogies	best model in syntactic analogies (58.7)	300
Wang2Vec	good in POS tagging	best model in POS tagging (95.94)	1000
GloVe	good in semantic analogies	best model in semantic analogies (48.5)	600

In [BGJM17] the authors extracted all n-grams for $3 \leq n \leq 6$, and represent each word by the sum of the representation of its n-grams. The main feature of this model is the capacity to generate a representation for words that were not seen during training. It is shown that FastTest outperforms W2V's CBOW in the English Rare Words dataset and in word similarity datasets of languages with rich morphology, such as German and Russian. While FastText outperforms W2V's CBOW in analogies based on syntax, W2V remains better in semantic analogies.

Word2Vec is really good in learning semantic relations, however, it is not that good in syntax, an important feature if the downstream task is a Part-of-Speech (POS) tagger or a Dependency Parser. This motivated [LDBT15] to propose the Wang2Vec model. In this model word representations are concatenated so that their position in the phrase may be discriminative. Through this change the model was able to increase in 1% the accuracy of the POS tagger and of the Dependency Parser. Wang2Vec and FastText are not compared in the papers, neither are they tested in the same tasks.

The Global Vectors (GloVe) model tries to conciliate a matrix of co-occurrence of words (size $T \times T$) with a fast training model in order to produce a word vector space of meaningful sub-structure. Using 300 dimensions and a corpora of 42 billions tokens, GloVe outperformed W2V in both semantic and syntactic analogies, scoring 81.9% and 69.3% respectively. The FastText model achieved 77.8% and 74.9% in the same task (in English), using only a dump of Wikipedia articles. For comparison, GloVe learned using the Wikipedia and vectors of size 100 achieved 67.5% in semantic analogies and 54.3% in syntactic analogies, and when using vectors of size 300 achieved 80.8% in semantics and 61.5% in syntactic, this way winning in semantics.

In [HFS+17] the authors present a comparison of Word Embedding trained on the same data: a heterogeneous Portuguese corpora of 1.5 billion tokens taken from a 2016 Wikipedia dump, news crawled from GoogleNews, magazine articles, movies' subtitles, etc. The embedding compared are: Word2Vec, Wang2Vec, FastText — these three considering both CBOW and Skip-Gram models — and GloVe, each having vectors of size 50, 100, 300, 600 and 1000. These vectors were made available[3] and are the ones we use in this work. The best results in semantic and syntactic analogies were achieved, respectively, by 600-sized GloVe (48.5%) and 300-sized FastText's Skip-Gram (58.7%).

In Table 2.1 we present a comparison between what was expected from the models — the tasks in which they should perform well — and what they achieved in the tests conducted in [HFS+17]. The expectations of each model were fulfilled in Brazilian Portuguese (PTBR), but the accuracy of the models in Portuguese and in English remain very distinct.

[3] http://www.nilc.icmc.usp.br/nilc/index.php/repositorio-de-word-embeddings-do-nilc

Chapter 3

Solving Standardized Tests

In this chapter we review the literature on solving standardized tests. We review attempts to solve the New York Regents 4th Grade Science Test, the Japanese National Center Test for University Admission and the Chinese Gaokao. The first exam is solved by two algorithms: Praline and Aristo; we focus our description in two components of Aristo, as they are used in this work. The second exam is used mostly as database to other tasks and we focus on its machine-readable structure. Finally, we describe the Information Retrieval strategy used to solve the third exam.

3.1 4th Grade Science Test — Praline and Aristo

The New York Regents 4th Grade Science is an American standardized test of multiple-choice questions written in simple English on Primary School Science topics. These topics generally require a great deal of commonsense reasoning and logical reasoning. Here we review two algorithms that try to solve this test, Praline [KBG+15] and Aristo [CEK+16]. Examples of questions of this test are:

1. Which gas is given off by plants? (A) Hydrogen (B) Nitrogen (C) Oxygen (D) Helium.
2. A mother hen clucks loudly when danger is near and her chicks quickly gather around her. Which sense helps the chicks receive this warning about danger from their mother? (A) smell (B) taste (C) sight (D) sound.

The first of the systems, Praline, uses Markov Logic Networks (MLN) to reason over knowledge represented in First Order Logic (FOL). MLN are defined as a set of pairs (F_i, w_i) — where F_i is a FOL formula and w_i is a real number — and a set of constants $C = \{c_1, ...c_{|C|}\}$.

MLN is a way to unite FOL with probabilities, it can be seen as a way to soften the constraints of FOL. When using a knowledge base in FOL any world that violates a formula has probability 0, the idea of MLN is to soften this constraint such that a world violating a formula becomes less probable, but still possible. The strategy is to attribute to each formula a weight, higher weights penalize more the violation of that formula [RD06]. The authors argue that they chose MLN because it can handle potentially cyclic rules.

The knowledge base (KB) of this algorithm comes from fourth grade science textbooks augmented with a web-search of the terms appearing in these texts. The authors state that most of the knowledge is in IF-THEN format and was extracted using handcrafted rules. They also add lexical reasoning probability and uncertainty in derived rules. For instance, they have a representation that cat and feline are synonyms with some probability. The weights of the formulae were set manually.

The MLN approach is used to solve the exam by translating each question of *k-alternatives* into *k true-false* questions $Q_{i\in(1,...,k)}$ that are either false or true with some probability. Then, they seek the most likely answer option: $arg\ max_{i\in(1,...,k)}\ Pr(Q_i|KB)$, where KB stands for Knowledge Base. Using this approach the algorithm achieved 47.5% accuracy.

Aristo, on the other hand, uses five different solvers to solve the same exam. These five solvers are divided into three different layers, namely, Text as Knowledge, Statistical Knowledge and Structured Knowledge. The first layer contains one solver, based on Information Retrieval; the second layer contains two solvers: one based on Pointwise Mutual Information and other based on SVM; the third layer also contains two solvers, one identical to Praline and the second solver is based in Integer Linear Programming.

The first solver searches for a document containing the question and a candidate-answer, repeating the process to each candidate-answer and selecting the one with highest scoring. More formally, they approach answering multiple-choice questions as an IR problem by using as query search the concatenation of the text from *question* plus an *and* operator and the text from *candidate-answer*. This operator ensures that the documents retrieved have at least one word of *question* and one word of *candidate-answer*. The second step is to retrieve a set of documents and pick the score of the highest scoring — the Score function, in this case, becomes a "belief measure" of the corpus on that query. This is repeated for each alternative, having in the end a belief measure for each alternative. They expect that the correct answer is the one with highest belief, being this the criteria to chose the final answer. This solver achieves 60.6% accuracy in the exam.

The second layer contains two solvers that attempt to identify connections between the question and candidate-answer: the Pointwise Mutual Information based solver looks for statistical association between words from question and candidate-answer. Alternatively, the Support Vector Machine (SVM) based solver — an implementation of the lexical semantics model presented in [JSC14] — looks for similarity between words of the question and words of the alternative. This is done by training four different groups of Word2vec, each in a different corpus. Then these vectors are used to extract, per group, two features that will be fed to a SVM. The first feature is the average of the cosine similarity between each word of *question* and each word of *candidate-answer*; the second feature is the cosine similarity between the vectors v_q and v_a, the first is the sum of the (vectors of) words of *question* normalized to have unit length. Similarly, the second vector is the sum of the words of *candidate-answer* also normalized to the unit length. The SVM is trained to answer if the 8 features — two of each group of vectors — represent a correct answer or not. This SVM Solver achieves 55.4% accuracy in the test.

The last layer uses two solvers that attempt to derive facts from domain specific general truths. The first solver is the same used by Praline, namely the MLN solver; the second — TableILP — structures knowledge in tables and joins them to derive conclusions. MLN scores 54.3%, TableILP 43.8%. Each of the five solvers returns a score in each candidate-answer. The final phase of Aristo is combining these scores using two logistic regressions. By doing so it achieves 71.3% accuracy, showing that combining different algorithms' scores leads to better results than using the algorithms individually.

One point that we must stress is that this test has four candidate-answers, while ours has five. This makes the comparison in accuracy not so direct. Finally, we point out that the same authors in a recent work [CCE+18] present a new dataset of 7800 science questions — also of 4 candidate-answers. These questions are divided into two groups: "Easy" and "Challenging". Although they do not use Aristo directly on this dataset, the IR solver got 62.55% and 20.26% accuracy in Easy and Challenging, respectively. TableILP achieved 36.15% and 26.97%. This suggests that Aristo's approach might not generalize well to different exams, even if related to the same domain — Scientific knowledge.

3.2 National Center Test

The Japanese National Center Test — hereafter Center Test — was proposed as benchmark for Natural Language Processing by Miyao and Kawazoe [MK13]. This multiple-choice test consists of questions about eleven subjects written in Japanese in addition to Foreign Language questions, written in English. This test is used by universities in Japan as an admission test and covers the following subjects: World History, Japanese History, Politics & Economics, Ethics, Physics,

Chemistry, Biology, Mathematics, Japanese, English and Modern Society.

While the exams layout is highly organized, automatically recognizing the structure is not trivial. This led the authors to created a XML version of the exams containing annotations or tags. These annotations describe the structure, the question and some linguistic information.

For structure annotation the authors intend to show that a fragment contains a question, or that the section contains an instruction or some fragment of data, etc. The structure tags used are shown in Figure 3.1. Similarly, there are tags that point out what kind of sub-task a question requires. Each question can be viewed by two perspectives: the type of answer and the type of knowledge it requires; these types are shown in Figure 3.2. Linguistic tags are those that appoint technical terms, dependency trees and co-reference relations.

Tag	Denotes
question	A question region including outermost question areas and minimal areas. An ID is assigned to each element. Question regions that do not include other question regions are given the attribute *minimal="yes"*, indicating smallest units of questions.
instruction	A statement or an instruction for a question.
data	Data provided to test-takers of reference, including not only texts but also images, tables, graphs, etc.
label	A label such as section numbers, question numbers, identifiers of text fragments, etc.
ansColumn	An identifier of an answer column. Each answer column is given an unique ID, which is referred to in answer data.
choices	A set of choices.
choice	An individual choice. The attribute *ra="yes"* denotes correct choices.
cNum	An identifier of a choice.
ref	A symbol that refers to another text fragment, such as underlined texts. A referred text fragment is denoted by the attribute *target*.
uText	An underlined text fragment. A unique ID is assigned when the text fragment is reffered to by *ref*.

Figure 3.1: *Types of structural annotations. Taken from [MK13].*

Answer Types	
sentence	Choices are described by sentences.
term	Choices are described by terms.
image	Choices are represented by images.
formula	Choices are represented by formulas
combination	Choices are described by a combination of sentences, terms, etc.
Knowledge types	
KS	An external knowledge source is required.
RT	Reading comprehension of a text given within a question is required.
IC	Image Comprehension is necessary.
GK	General Knowledge is required.
DM	Domain-specific inference is required.

Figure 3.2: *Types of question annotations. Taken from [MK13].*

In [MMS+14] the authors describe the RITE competition, in which systems were evaluated by their capacities of recognizing semantic relations between texts t_1 and t_2. That is, by reading t_1 what is possible to state about t_2? The possible answers are: t_1 entails t_2, t_2 contradicts t_1 or it is not possible to relate them. The Center Test database was used in this competition, which had a version in Japanese, Chinese and English.

In the Chinese competition only t_2 was given — taken from the database — and the systems had to find in Wikipedia a t_1 that entails or contradicts t_2. The questions were taken from Physics, Chemistry and Biology, while questions of History and Geography were adapted, because they were easier to solve using Wikipedia. All questions were rewritten in a declarative sentence.

The Japanese competition contained not only Wikipedia, but also textbooks as knowledge base. The questions involved World History, Japanese History and Politics & Economics. The questions were modified in order to transform the candidate-answers into closed sentences in t_2. The same modifications were done in the English competition, whose questions considered World History and Politics & Economics.

The winner algorithm of the Chinese competition achieved 44% accuracy, having 45.22% precision in entailment and 31.67% precision in contradiction. The winner of the Japanese competition achieved 63.23% accuracy and the English winner had 55.85% accuracy.

In [SSK+14] the Center Test was used in a Question Answering competition where a corpora of textbooks and a version of the Wikipedia were provided as knowledge base. The questions of this competition were taken from World History subject. The authors state that these questions are not simple QA format, requiring understanding of the surrounding context and inference. The answers could be true/false or short statements. To adapt the QA format to the questions database, the competition provided a QA system that has two additional layers: the first is responsible for analyzing the type of question (true/false or statement) and the second for choosing the alternative based on the answers generated and their scores. The winning team achieved 77% accuracy using deep learning and case-frame graphs as semantic representation obtained by parsing textbooks using a semantic parser. This algorithm did not perform well in short answer questions.

In [FKKM14] the authors review the achievements of the Todai Project, which aims at having a robot accepted in Tokyo University by 2021 through the Center Test. There was a competition where task-takers could attempt to solve exams about one of the following subjects: English, Japanese, Japanese History, Math IA, Math IIB, Physics and World History. We remark that Mathematics is the only subject that requires constructed (open) responses, the answer is a numerical value. They also applied a test for students who would take the Center Test that year. The result of the highest scoring algorithm in each subject is presented in Table 3.1 in comparison with the average of the students.

Table 3.1: *Comparison between best scoring algorithm and average of students. Taken from [FKKM14].*

Scores	English	Japanese	Jp. Hist.	Math I	Math II	Physics	World Hist.
Best system	26%	41.3%	56%	57%	41%	39%	58%
Student's Average	44.1%	48.1%	45.6%	52%	47.6%	42%	46.6%

The authors state that there were two strategies to solve Japanese History and World History: trying to recognize entailment and using a QA Engine, which was more successful. In these two subjects the score of the best algorithm was higher than the students' average. The entailment approach was used also in English, but in this subject commonsense reasoning was required — in order to choose the most adequate word in a dialogue or sentence —, consequently its performance was not so good. In fact, the best and the worst performances come from this approach. In Mathematics the strategy was to derive first-order predicate logic formulas from high-order predicates logic formulas and solve the problem using formula manipulation algorithms. The authors state that most questions that give a formula and some instructions in natural language were correctly solved using this method, but questions fully stated in natural language remained a hard problem. Similarly, in Physics they used a physical simulator, but it often requested a parameter not informed in the statement and thus had unexpected behaviors. The authors also remark that several strategies were used and that the question type was determinant to choose a strategy.

3.3 Gaokao

The Gaokao is an exam applied annually in China and is the world's biggest entrance examination in number of registered candidates. The exam comprises three mandatory subjects and other three from a group that the task-taker must choose: either Social Science or Natural Science. The mandatory subjects are: Chinese, Foreign Language and Mathematics; Social Science comprises History, Geography and Politics; Natural Science is composed of Chemistry, Physics and Biology.

A dataset containing multiple-choice questions of History taken from this exam was made available and solved using Information Retrieval [CZW+16]. The authors divided the dataset in two groups: one containing 121 questions which could be answered using the content of Wikipedia, and the second containing 454 questions that can not be answered using uniquely the Wikipedia, requiring textbooks to be answered. The authors argue that often questions of History rely on identifying/retrieving the meaning of an idiomatic expression that is written in an older version of Chinese (Classical Chinese), in other words, that is not written in current ideograms (Simplified Chinese).

To solve these questions the authors retrieve articles from the Chinese Wikipedia using their title. The pages retrieved using words of the statement become either Concept Pages or, in case the words are between quotation marks in the original statement, Quote Pages of the Statement. The same is applied to each alternative independently, creating Concept Pages and Quote Pages for each alternative. These retrieved pages have their tf-idf summed to create a representation of what is known. Each alternative has its representation evaluated by the cosine similarity with the representations of the statement, yielding two measures: the similarity of Concept and the similarity of Quote. The authors combined these two measures by summing them, thus achieving 43% and 31% accuracy in the first and second groups, respectively. All questions of this dataset have four candidate-answers, consequently a random guesser is expected to achieve 25% accuracy.

The approach used by the proponents relies on two points: quotation and idiomatic idiosyncrasies. Other exams do not necessarily require or present quotations; furthermore, the fact that quotes are written using unusual ideograms benefits Information Retrieval, thus the performance of the heuristic used by the authors may be lower in other idioms.

Chapter 4

Creating the ENEM Dataset

In this chapter we start by describing in Section 4.1 the exam we are using to create the dataset. Then we present its translation into a machine-readable format in Section 4.2. The structural and informative tags are presented in Section 4.3 and the relation of multiple knowledge tags is presented in Section 4.4; Finally, in Section 4.5, we present how many questions that are for each tag.

4.1 The Assessment

The *Exame Nacional do Ensino Médio (ENEM)* is taken by the majority of Brazilian students who wish to enroll in an undergraduate education program. Its objective part consists of 180 multiple-choice questions evenly split into four major topics: Humanities, Language, Sciences and Mathematics. The Language part includes five questions on a Foreign Language — either English or Spanish, depending on the student's choice —, which we discarded in order to simplify matters. The exam usually takes two days; from 2010 to 2016 questions on Humanities and Sciences took place on the first day, while Language and Mathematics took place on the second day; in 2009 and 2017 there were changes in the combinations, but the fact that in each day two major topics are applied did not change.

Digital copies in PDF of the previous editions of the exam along with their solutions, can be freely downloaded from the official website[1]. Each exam contains a mixture of images and text in a loose pattern. An example is given in Image 4.1, where two pages are presented: the page on the left is divided into two columns, while the page on the right has a single column. Questions usually have the following format: a text or image is presented (the header), followed by a textual statement, then five candidate-answers are given, one of which is correct. Sometimes a header is shared by two questions.

4.2 XML Structure

In order to maintain the focus of this research on knowledge processing we created a machine-readable dataset of questions by manually converting exams into structured textual form (XML format). We only retained the textual part of questions and discarded any non-textual information, although we did use it to annotate questions, as will soon be explained. We segmented each question into three parts: the *header*, containing a text fragment given as base knowledge or context for the question; the *statement*, which contains the statement of the question, that is, what is being asked; and the *candidate-answers* or *alternatives*, which contains the text of each answer-candidate and flags the correct choice. If a header is shared by more than one question we clone it and put it in the header of each question to which it refers.

Figure 4.2 depicts an example of a digitalized question and its machine-readable format in

[1] http://inep.gov.br/provas-e-gabaritos

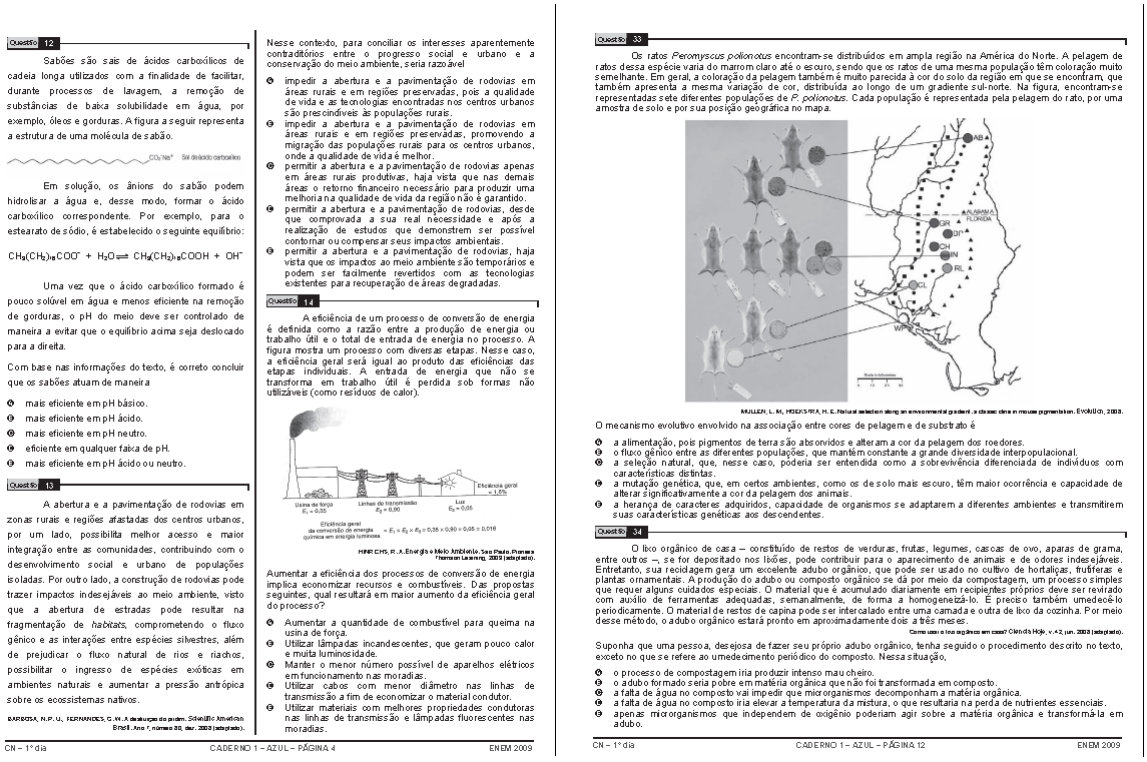


Figure 4.1: *Example of two pages of multiple-choice test of ENEM. Note how images, text and formulas are disposed without a pattern.*

the structured form we propose[2]. The picture contains a text written by a student describing the Brazilian flag, sketched on the upper left corner. Textual information that is irrelevant or not easily recognized as text is, at this stage, ignored from the conversion. In the example, this includes the reading of the stamp mark and the annotations in the picture.

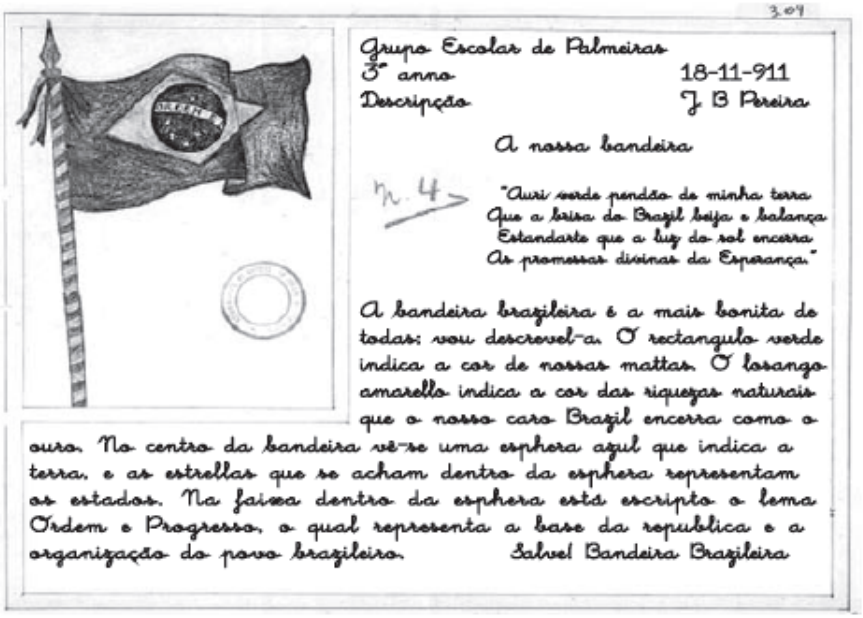
In order to help with the performance analysis of the techniques we associate informative tags to the questions. The tag "image" (IMG) is associated to every question that is accompanied by an image, regardless of whether the image is actually important/crucial to answer the question. By image we considered anything that is not purely textual: drawings, pictures, graphics, tables and diagrams. The remaining tags inform the kind of knowledge (or tasks) that is (in principle) necessary to answer the question. We stress that these tags are general because we do not want tags to be so fine-grained that the questions with the same tag are almost identical, allowing algorithms to be created specifically for that tag.

4.3 Knowledge Tags

The tag "encyclopedic knowledge" (EK) suggests that the question resembles (or is) a factoid question, thus it can be answered by consulting an external source of knowledge such as an encyclopedia. This is in contrast with questions that can be answered only using the text or image (through commonsense knowledge and reasoning). Examples of questions tagged as EK include the characteristics of an epoch, social movement and the the main ideas of a philosopher. An example of this type of knowledge is given in Figure 4.3 — hereafter we abstract the XML syntax and present questions in a tabular structured format.

The tag "image compreheension" (IC) is assigned to questions which require identifying or

[2]We did not translate this question because it is shown in its original version (pdf) and we thought it could be misleading to present the XML version in English, one could think that we translated the questions of the dataset, which we did not. Additionally, the focus is to show that the non-textual information is ignored in the machine-readable version.



3.09

Grupo Escolar de Palmeiras
3º anno 18-11-911
Descripção J. B Pereira

A nossa bandeira

n. 4

"Auri verde pendão de minha terra
Que a brisa do Brazil beija e balança
Estandarte que a luz do sol encerra
As promessas divinas da Esperança."

A bandeira brazileira é a mais bonita de todas; vou descrevel-a. O rectangulo verde indica a cor de nossas mattas. O losango amarello indica a cor das riquezas naturais que o nosso caro Brazil encerra como o ouro. No centro da bandeira vê-se uma esphera azul que indica a terra, e as estrellas que se acham dentro da esphera representam os estados. Na faixa dentro da esphera está escripto o lema Ordem e Progresso, o qual representa a base da republica e a organização do povo brazileiro. Salve! Bandeira Brazileira

GRUPO ESCOLAR DE PALMEIRAS. Redações de Maria Anna de Biase e J. B. Pereira sobre a Bandeira Nacional. Palmeiras (SP), 18 nov. 1911. Acervo APESP. Coleção DAESP. C102 79. Disponível em: www.arquivoestado.sp.gov.br. Acesso em: 15 maio 2013.

O documento foi retirado de uma exposição *on-line* de manuscritos do estado de São Paulo do início do século XX. Quanto à relevância social para o leitor da atualidade, o texto

(A) funciona como veículo de transmissão de valores patrióticos próprios do período em que foi escrito.
(B) cumpre uma função instrucional de ensinar regras de comportamento em eventos cívicos.
(C) deixa subentendida a ideia de que o brasileiro preserva as riquezas naturais do país.
(D) argumenta em favor da construção de uma nação com igualdade de direitos.
(E) apresenta uma metodologia de ensino restrita a uma determinada época.

<question CE="No" DS="No" IC="No" EK="No" MR="No" TC="Yes" id="96" image="Yes">
<header>
Grupo Escolar de Palmeiras 3º anno 18-11-911 J B Pereira Descripção A nossa bandeira “Auri verde pendão de minha terra Que a brisa do Brazil beija e balança Estandarte que a luz do sol encerra As promessas divinas da Esperança.” A bandeira brazileira é a mais bonita de todas; vou descrevel-a. O rectangulo verde indica a cor de nossas mattas. O losango amarello indica a cor das riquezas naturais que o nosso caro Brazil encerra como o ouro. No centro da bandeira vê-se uma esphera azul que indica a terra, e as estrellas que se acham dentro da esphera representam os estados. Na faixa dentro da esphera está escripto o lema Ordem e Progresso, o qual representa a base da republica e a organização do povo brazileiro.
</header> <statement>
O documento foi retirado de uma exposição on-line de manuscritos do estado de São Paulo do início do século XX. Quanto à relevância social para o leitor da atualidade, o texto
</statement> <answers>
<option id="A" correct="Yes"> funciona como veículo de transmissão de valores patrióticos próprios do período em que foi escrito. </option>
<option id="B" correct="No"> cumpre uma função instrucional de ensinar regras de comportamento em eventos cívicos. </option>
<option id="C" correct="No"> deixa subentendida a ideia de que o brasileiro preserva as riquezas naturais do país. </option>
<option id="D" correct="No"> argumenta em favor da construção de uma nação com igualdade de direitos. </option>
<option id="E" correct="No"> apresenta uma metodologia de ensino restrita a uma determinada época. </option>
</answers>

Figure 4.2: *Example of original question (top) and machine-readable format (bottom). Some text was suppressed from the header to save space.*

Tags
EK
Header
The evolution of the transformation of raw materials into finished had three stages: handicraft, manufacture and machinofacture.
Statement
One of the stages was the handicraft, in which
Alternatives
(a) people worked as the rhythm of the machines and in a standardized way. (b) people worked usually without using machines, differing from the assembly line model. [correct] (c) people used abundant power sources to make machines work. (d) each worker did part of the production, machines were used and the workers received salaries. (e) technicians and managers determined the production rhythm by interfering in the production process.

Figure 4.3: *Example of question tagged as Encyclopedic Knowledge. Our translation.*

understanding the constituent elements of a given painting, cartoon, photo or advertisement. We remark that we included in this class images that contain text, even when it is only this text that might be crucial to produce the correct answer. An example is a question that displays a cartoon and then asks: "The cartoon criticizes the means of communication, specially the Internet, because", this one demands understanding what is inside this cartoon, therefore the answer lies in the image. Image Comprehension is also used for some interpretations of mathematical graphics, such as identifying points of interest, like points of maximum or minimum of a given graphic: the month with most sales, with minimum loss, etc.

A question is tagged as "text comprehension" (TC) if the answer can be identified somehow using the given text. As the answer almost never is stated *ipsis litteris*, questions with this tag tend to require some sort of reasoning about what is stated and also, frequently, ask for identifying: (1) the author's thoughts or feelings; (2) figures of speech and/or (3) passages with some characteristics. These are usually highlighted by the presence of expressions such as "as the author", "present at the text fragment", etc. We present in Figure 4.4 an example of question with this format. We point out that questions tagged as TC are the ones that most require commonsense reasoning and understanding of (Brazilian) Portuguese.

Note that if no graphical feature — e.g., font, text layout, etc. — is relevant to producing the answer, then the question is tagged as TC and not as IC. An example is the one in Figure 4.2, where the graphical features play no role in both statement or answers.

For the exams of Sciences and Mathematics, it would be useful to identify two types of knowledge: (1) one stating the necessity to (understand and) convert the given problem from natural language into a mathematical or chemistry formula, solve it and from this answer identify the correct candidate-answer; (2) one identifying that the question requires finding and understanding domain specific rules or that complex inferences are required; for domain specific rules we understand the Laws of Physics, Thermodynamics, Mass Conservation, Genetics and so on. We call this first type of knowledge Mathematical Reasoning (MR) and the second of Domain Specific (DS). An example of each is given in Figure 4.5 and in Figure 4.6, respectively.

Some questions of Physics, Mathematics and Chemistry present complex formulae. To represent them in our XML dataset we did as [MK13] and transformed the formulae into MathML notation. Some examples are shown in Table 4.1. By using MathML it is possible to represent both Mathematical, such as the first 3 rows, and Chemical formulae, such as the last 3 rows. It is also possible to differentiate between variables and text, represent superscript and subscript, among other desirable representations. We marked with a special tag, Chemical Element (CE), if a question requires any knowledge or special treatment — that is, it can not be treated simply as text — over a given element.

Tags
TC
Header
TEXT I Our fight is for the democratization of land property, which is getting more and more concentrated in our country. Around 1% of all landowners controls 46% of the land. We pressure through occupations of big or unproductive land properties, that don't do their social part, as the Constitution of 1988 demands. We also occupy farms whose land was stolen from public land. TEXT II The small landowner is equal to a small store owner: the smaller the business, harder it is to keep it running, because the charges are heavy and it must profit. I am in favor of productive and sustainable properties that generate jobs. Supporting a productive, job generating enterprise is cheaper and generates much more than supporting land reform.
Statement
In each fragment the authors oppose each other. This happens because the authors associate the land reform, respectively, to
Alternatives
(a) reduction of city swelling and criticism on small land owners. (b) growth of national funds and prioritize the international market. (c) stopping the mechanization of agriculture and fighting the rural exodus. (d) privatization of state companies and economical growth stimuli. (e) correcting historical distortions and loss of agribusiness. [correct]

Figure 4.4: *Example of question tagged as Text Comprehension. References were suppressed on the example, our translation*

Tags
MR
Header
A factory produces parallelepiped and cubic shaped chocolate bars, both with the same volume. The parallelepiped bar has edges of 3 cm width, 18 cm height and 4 cm of depth.
Statement
Given the characteristics of the geometric figures presented, the size of the edge of the cubic chocolate bar is:
Alternatives
(a) 5cm. (b) 6 cm. [correct] (c) 12 cm. (d) 24 cm. (e) 25 cm.

Figure 4.5: *Example of question tagged as Mathematical Reasoning. Our translation*

Tags
DS
Header
One of the environmental problems of the nowadays agriculture is the soil compaction, due to the intense traffic of machines increasingly heavy, reducing the productivity of the cultures.
Statement
One way to prevent soil compaction is substituting the tractor's tires by
Alternatives
(a) larger tires, reducing the pressure on the soil. [correct] (b) narrower tires, reducing the pressure on the soil. (c) larger tires, increasing the pressure on the soil. (d) narrower tires, increasing the pressure on the soil. (e) higher tires, reducing the pressure on the soil.

Figure 4.6: *Example of question tagged as Domain Specific. Our translation*

Table 4.1: *Examples of formulae into MathML.*

Formula	MathML
y=5x	<mi>y</mi><mo>=</mo><mn>5</mn><mi>x</mi>
IMC = 20	<mtext>IMC</mtext><mo>=</mo><mn>20</mn></math>
15-3x	<mn>15</mn><mo>-</mo><mn>3</mn><mi>x</mi></math>
CO_2	<msub><mtext>CO</mtext><mn>2</mn></msub></math>
Na^+	<msup><mtext>Na</mtext><mo>+</mo></msup>
Na_2^+	<msubsup><mtext>Na</mtext><mn>2</mn><mo>+</mo></msubsup>

Tags
TC EK
Header
Six p.m., Preciados Street. Far away, the human mass that fills the Puerta Del Sol Square in Madrid stands up. A group of girls, seeing this, runs towards the crowd. Millions of people shout the slogan: "Do not, do not, do not represent us". A boy speaks in the megaphone: "We demand a referendum about the bailout".
Statement
In 2011, the Spanish Indignados' encampment expressed the discontent of the European youth with the politicians. Which proposal synthesizes the set of political claims made by these young people?
Alternatives
(a) Universal Suffrage. (b) Direct Democracy. [correct] (c) More political parties. (d) Autonomous legislation. (e) Parliamentary immunity.

Figure 4.7: *Question requiring text comprehension (TC) and encyclopedic knowledge (EK), our translation.*

4.4 Multiple Knowledge Tags

It is necessary to remark that the knowledge tags are not mutually exclusive. For example, consider the question in Figure 4.7. The correct answer is "direct democracy", which requires both interpreting the text, understanding its context and knowing basic facts.

Questions tagged as IC and EK usually present an image of an event or person mentioned in the statement; questions on cartoons are usually TC and IC, since the text appears inside an image and the answer is in textual form.

The combination DS and IC or TC appears frequently when the header has a description/image of a situation and the statement asks for the consequences or what can be done to avoid this situation. DS and EK are mutually exclusive, because DS already has this intuition of searching for knowledge outside what is given in the question. Questions that require finding a mathematical formula and applying in the context given are tagged as both DS and MR.

The MR tag does not occur frequently with the others. In fact, in only one question did MR and EK co-occur, the question addressed the reduction in the emission of SO_2, but giving all the data to solve the question. The exception is the IC tag, which often co-occurs with MR, since we treat graphics and tables as image, and when they are given they are fundamental to answer the question.

4.5 Characteristics of the ENEM Dataset

Table 4.2 shows the overall number of questions in the exams of Humanities (1) and Languages (2) discarding Foreign Language questions from 2009 to 2017, as well as the number of questions associated with each tag. We see that (i) most questions require text comprehension, (ii) about 40% of these questions require/can be answered by consulting an external knowledge base, and (iii) there are many questions that use irrelevant images. Note that in the dataset there is no tag of exclusivity, hence TC_{only} and EK_{only} are shown for analysis purposes only. We omitted DS and MR because they occurred less than 5 times in these groups during this period of time.

Table 4.2: *Usage of each type of knowledge on Humanities(1) and Languages(2) from 2009 to 2017. * - especially in 2016 there were two applications*

EXAM	#TOTAL	IMG	TC	EK	IC	TC_{only}	EK_{only}
2009-1	45	10	26	30	10	10	13
2009-2	44	14	40	10	14	22	1
2010-1	45	5	31	29	1	16	13
2010-2	40	9	31	9	8	25	3
2011-1	45	9	30	32	6	12	11
2011-2	40	12	29	11	11	21	2
2012-1	45	9	31	20	6	21	8
2012-2	40	13	35	11	10	23	3
2013-1	45	10	32	20	9	19	5
2013-2	40	12	35	10	10	23	0
2014-1	45	12	32	28	10	13	7
2014-2	40	8	34	12	7	22	3
2015-1	45	10	36	18	7	22	4
2015-2	40	11	35	9	8	23	1
2016-1	45	10	38	23	7	18	3
2016-2	40	6	37	7	4	30	1
2016-1(2)*	45	12	32	29	10	14	4
2016-2(2)*	40	10	34	14	8	20	3
2017-1	45	6	39	21	4	18	0
2017-2	40	12	40	11	10	29	1
Total	854	200	677	354	160	401	86

Table 4.3 shows the number of questions in the period of 2009 to 2017 and the number of questions that use each kind of knowledge in the exams of Sciences (3) and Mathematics (4). It is possible to see the dominance of MR and IC in the exams of Mathematics, due to the fact that most questions require understanding a table or graphic and also translate natural language into formula. DS is dominant in Sciences and occurs frequently with the other types of knowledge.

To sum up, we created an XML dataset of questions taken from exams between 2009 and 2017. This dataset contains 1755 questions, each labeled according to the type of knowledge necessary to solve it: Text Comprehension, Image Comprehension, Encyclopedic Knowledge, Mathematical Reasoning and Domain Specific.

Table 4.3: *Usage of each type of knowledge on Sciences(3) and Mathematics(4) from 2009 to 2017. * - especially in 2016 there were two applications*

EXAM	#TOTAL	IMG	TC	EK	IC	DS	MR
2009-3	45	23	19	14	17	37	9
2009-4	45	30	0	0	23	1	42
2010-3	45	19	22	15	13	40	9
2010-4	45	31	0	0	31	2	40
2011-3	45	21	17	19	15	35	3
2011-4	45	26	0	0	24	0	42
2012-3	45	20	25	13	15	40	8
2012-4	45	30	2	0	26	0	42
2013-3	45	22	8	13	19	42	12
2013-4	45	34	0	0	27	0	33
2014-3	45	22	21	16	22	41	4
2014-4	45	29	0	0	23	0	43
2015-3	45	25	21	12	21	27	13
2015-4	45	25	0	0	20	0	43
2016-3	45	19	19	16	17	37	15
2016-4	45	27	00	0	25	0	41
2016-3(2)*	45	10	29	16	12	36	10
2016-4(2)*	45	25	01	0	22	0	41
2017-3	45	23	22	15	20	34	14
2017-4	45	30	00	0	26	1	42
Total	900	491	206	149	418	373	506

Chapter 5

Solving the ENEM

In this chapter: we describe, in Section 5.1, how we use Information Retrieval to answer multiple-choice questions; in Section 5.2 we present how we use Word Embedding with the same goal. Section 5.3 presents how WordNet and Word Embedding are used to aggregate some (superficial) knowledge to the two initial approaches. Finally, in Section 5.4, we present the greedy search to find an ensemble of algorithms that together are better than individually.

5.1 Information Retrieval

One of the first strategies one may think of when trying to solve a multiple-choice question is to look for keywords that appear both in the statement and text of a candidate-answer. Consider for example the question in Figure 5.1: one should identify promptly that the second item of the Header has no relation with the Statement, remaining only the first and third statements. Latin and Greek do not appear in the candidate-answers, so they can be ignored, consequently we can answer correctly by identifying that Goethe and German appear in the same text; it is not even necessary to know that Die Leiden des jungen Werthers equals The Sorrows of Young Werther or that J. W. stands for Johann Wolfgang.

In order to solve questions using IR we follow Aristo and use Lucene[1] to index and retrieve documents — what will be indexed will depend on the strategy/heuristic being used, as will be discussed soon.

In our dataset the questions' header frequently have a text that serves as base for the statement, consequently there must be some information in the header that is useful. One of our strategies is to identify which document composed by candidate-answer plus statement is most similar to the

[1]We used Apache Lucene version 6.4, since then the software received some upgrades and the actual version is available at: https://lucene.apache.org/core/downloads.html

Header
1) Johann Wolfgang von Goethe is the writer of the German classic Die Leiden des jungen Werthers.
2) The Japanese Miyamoto Musashi never lost a duel.
3) Shakespeare, Goethe and Schopenhauer knew both Latin and Greek.
Statement
J. W. Goethe wrote The Sorrows of Young Werther in:
Answers
a) English b) Japanese c) French d) German [correct]

Figure 5.1: *Example of question solvable by word look up.*

header. This is our Information Retrieval Header strategy or, for short, IR-H. More formally, being d_s the document containing the statement, d_a, d_b,..., d_e the documents containing the candidate-answers from a to e, respectively, and d_h the document containing header:

$$Database_{IR-H} = [d_s + d_a,\ d_s + d_b,\ d_s + d_c,\ d_s + d_d,\ d_s + d_e]$$
$$query = d_h$$

It is important to note that this strategy has *dynamic* knowledge base and query, meaning that they change for each question — this characteristic will be important when we discuss Augmentation in Section 5.3. The problems with this strategy are that seldom the answer is stated verbatim in the header and often none of the words of statement and candidate-answer appear in the header, making all alternatives score zero.

To mitigate these problems we created a second strategy named Information Retrieval Wikipedia (IR-W) by indexing articles from the Wikipedia dump of January 2018 (1.3GB of plain text). We considered each page a different document — $article_1$,..., $article_N$. While articles from Wikipedia provide a rich database, they lack correlation with the questions. We thus investigated a third strategy, Information Retrieval ENEM (IR-E), which indexes documents from the ENEM exams — not including the one being solved — composed of questions' header (d^i_h), statement (d^i_s) and text of the correct answer ($d^i_{correct}$). We select the highest scoring candidate-answer as the correct answer, where the score for each candidate-answer is the score of the search query formed by the conjunction ($\bigoplus$) of a document containing the statement and the text of the respective alternative. The conjunction makes the score of a document become zero if it does not have at least one word of statement and one of alternative.

$$Database_{IR-W} = [article_1,\ article_2,\ ...,\ article_N]$$
$$Database_{IR-E} = [d^1_h + d^1_s + d^1_{correct},\ ...,\ d^K_h + d^K_s + d^K_{correct}]$$
$$query = d_s \bigoplus d_{alternative}$$

These two strategies have *static* knowledge base, meaning that the set of indexed documents does not change depending on the question. Although this two last strategies eliminate the zero point scoring, the documents retrieved may be from very distinct subjects.

Additionally we developed two heuristics that try to combine the previous strategies: we call them Adding Heuristic (AH) and Non-Deciding Heuristic (NDH). Each of these has two variants: ENEM and Wikipedia, denoted by, for instance, AH-E or AH-W. The idea behind them is to combine IR-H with the others. By adding the score given by IR-H with IR-E we create AH-E — alternatively, AH-W is the (score) sum of IR-H with IR-W. NDH-W is defined as using IR-W when IR-H attributes zero to every alternative, NDH-E is using IR-E in place of IR-W. We interpreted the zero scoring of all alternatives as a state of "indecision", hence the name.

In [CEK+16] the authors achieved 60.6% accuracy using the same query structure as the described here when solving multiple-choice science questions without image. They tested their system in 129 questions using a knowledge base of 280 GB of plain text plus 80k sentences on elementary science, extracted from textbooks, study guides and crawled from the web. We remark the style difference between our questions and theirs: while questions from the ENEM are usually long, averaging three questions per page, elementary school Science questions usually are 8 to 70 words long.

The general limitation of the approach described in this section is the necessity of matching the words used in the query with the exact same words in the database. One approach to mitigate this problem is to augment the queries with similar or related words, this will be discussed in Section 5.3. A different approach is to adopt a different representation for documents, one that correlates documents that share semantics while differing on the exact words. We now turn to this approach.

5.2 Word Embedding

As discussed in Section 2.2, Word Embedding project words into a vector space such that words that occur frequently in similar contexts are closer than words that do not. We can thus use Word Embedding to compare documents.

The Word Embedding we use in this work are the ones made available by [HFS+17], namely, Word2Vec, Wang2Text, FastText — considering both CBOW and Skip versions — and GloVe, with vectors of dimensions 50, 100, 300, 600 and 1000. We use them in three manners: to solve directly the exams, to extract features that are used by a SVM, and to enrich text — this last usage is discussed in Section 5.3.

Intuitively, if a text document D_1 is semantically close to a text document D_2, then the sum of the vector representations of the words in D_1 should result in a vector close to the one obtained as the sum of the vector representations of words in D_2. We can thus solve a question by choosing the alternative whose vector representation, obtained as the sum of the vectors of its words, maximizes the cosine similarity to the vector obtained by summing the vector representations of the words in the description of the question. As we have statement and header describing the question, we tested two different heuristics: R1 and R2; in the first we treat D_1 as being the header and the statement, while in the second D_1 is just the statement.

The second usage of Word Embedding is to extract features based on D_1 and D_2. We extracted the same features presented in Aristo's SVM Solver: (1) the average cosine similarity between vector representations of words in D_1 with (vector representations of) words in D_2; and (2) cosine similarity between the sum of the vector representations of the words in D_1 and D_2 both normalized. We also tested adding two other features as presented by the same authors in [FJHP+15], namely, maximum and minimum similarity between words of D_1 with D_2, but the results were not sufficiently significant, so they are omitted.

5.3 Augmenting the exam

One strategy to increase the coverage of the IR approach is to augment the query and/or documents with additional words. We considered a method for augmenting the documents: we include words that meet a certain criteria, thus adding more context to the documents. The criteria to add a word to a document is that this word has some relation with another word already occurring in the document.

Ideally we would like a function that, given a word, returns other words related in some sense to the one given. Let us say that the related words are synonyms. We would like to augment an IR query using the synonyms of the words appearing in it, thus increasing the matching chance. That is, we would like a function ρ that takes a word w, a relation r and returns w plus a list L of words that have the relation r with the word w — being L empty if there are no words that match the relation:

$$\rho(r, w) = Lw$$

By using this function and WordNet[2] we can for a given document $D = w_1 w_2 w_3 ... w_n$, and s standing for "synonyms" augment D with the synonyms of every word by creating a new document D':

$$D' = \rho(s, w_1)\rho(s, w_2)...\rho(s, w_n)$$

The WordNet relations that we use are: (1) Synonym; (2) Hypernym; (3) Hyponym; and (4) Holonym. Additionally we also used (5) Contextual Co-occurrence, to identify words having this relation we use GloVe with 300 dimensions, we call this fifth type of GloVe augmentation. We took

[2]We used Open Multilingual WordNet, this WordNet aligns words in other languages to the English WordNet and uses the relations in English.

the vectors made available and computed the top ten closest words[3] of each word made available. We set the limit to ten words, but this value could be optimized. Finally, we did not limit the words from WordNet because, differing from GloVe, when WordNet outputs a synonym — for instance — for a given word, it does not state a value of how much of a synonym the output is, so it is not trivial to determine which or how much words we should use.

Although our initial intent was to only augment IR queries, we went beyond and augmented all the exams — the header, the statement and the alternatives. We use the five previous relations plus a special one: "Preprocessed", which pre-processes the text.

We differentiate the usage of each relation with an addition to the name of the method. For instance, IR-E-Hyponym should be read as: Information Retrieval using ENEM as database and modifying the given questions using Hyponym; Wang-CBOW-50-Normal-R1 should be read as Wang2Vec method using CBOW model with vectors of fixed size 50, using Normal text (without augmentation) and R1 heuristic; W2V-Skip-300-Hypernym-R2 as Word2Vec using Skip-Gram model with 300-sized vectors, text augmented using Hypernym and using the R2 heuristic.

In Table 5.1 we show variations of the fragment of text "apresenta uma voz lírica feminina que contrapõe o estilo de vida do homem ao modelo reservado à mulher" (presents a feminine lyrical voice that opposes man lifestyle with the one reserved to the woman). We emphasize using boldface the words in the non-augmented text. The words introduced by ρ applied on w are the words preceding w and succeeding the previous word in the original text — we suppressed from the table the offensive words returned. Additionally, we keep the text in Portuguese because we want to stress some problems we have when using WordNet and GloVe: consider the synonyms of "estilo" (style), there are words that do not fit, like "stylus" (stylus) and "modernidade" (modernity); the same applies to "homem" (man), which has "esposa" (wife) as synonym, and "mulher" (woman) which has "brasa" (fire) and "franguinha" (chick) — which may have been informal synonyms for woman in the past; in Hypernym we have a strange "servente" (servant) for "modelo" (model), and also "mulher" (woman) as hypernym of "homem" (man); the opposite happens with Hyponym: "marido" (husband), for instance, is hyponym for "mulher" (woman); in Holonym we can see that "casal" (couple) is a whole which "homem" (man) is a part; surprisingly, "mulher" (woman) does not have couple as holonym; GloVe finds some interesting relations: "restrito" (restrict) is close to "reservado" (reserved), but also finds undesired relations: "paz" (peace) as close word to "homem" (man) and "olímpica" (olympic) to "feminina" (feminine), to name a few.

A possible explanation for the undesired relations found using WordNet is that the word in Portuguese is translated to English and then the found words are translated back to Portuguese, which inserts naturally an error in the process. The problem of unfit close words of GloVe might be from bias of the texts used to train the model. Moreover, these methods for augmenting text consider each word separately; while it is possible that using a context sensitive augmenting procedure could lead to better results, we do not pursue this approach here.

Note that sometimes the WordNet outputs compound words, like "homo sapiens", "forma de vida" (life-form), "propriedades acústicas" (acoustic properties), etc., these words are given in Portuguese with an underscore connecting the words, we took the underscore off and kept the words.

It is crucial to stress that IR-E always uses the original (non augmented) version of the texts as database — although Lucene preprocesses it by lematizing and doing other operations. This means that when using IR-E augmented with Holonym (IR-E-Holonym) the queries contain the Holonym augmented version of the question, but the texts which will be searched are all in the Normal version; we also did not augment the texts extracted from the Wikipedia; IR-H, on the other hand, uses the text as it is given, so when using IR-H-Synonym the query is expanded and also the text with which it will try to match words is also expanded by the same relation. This is important because the heuristics Adding Heuristic and Non-Deciding Heuristic use combination of these two, so NDH-E-Hyponym will call IR-H-Hyponym. This means that the text given will be

[3]Closeness is measured in cosine similarity, we set the minimum threshold to be 0.5, if two words have less than this threshold they were not considered close. Consequently there are "isolated words", they do not have any close word.

Table 5.1: *The result of the different augmentations. In bold the words that appear on the original phrase.*

Variation:	Phrase:
Preprocessed	"**apresenta voz lírica feminina contrapõe estilo vida homem modelo reservado mulher**"
Synonym	" dispõe **apresenta** som voz humana **voz lírica** feminino **feminina contrapõe** estilete stylus elegância garbo modernidade maneira moda tendências espécie género tipo discurso língua linguagem **estilo** alma alacridade furor vida existência animação dia-a-dia dias **vida** homo sapiens humana humanidade humano ser humano espécie humana geração mundo povo macho varão namorado noivo garoto menino moço rapaz gajo cônjuge esposo marido homem pessoa colega cónjuge consorte esposa **homem** esquema ícone ídolo perfeição exemplo molde norma técnica padrões clichê gabarito predefinição manequim **modelo reservado** fêmea brasa franguinha garota gata menina moça senhorita amante namorada dama dona senhora esposa marido **mulher**"
Hypernym	"**apresenta** propriedades acústicas som voz voz humana comunicação mensagem cantador cantor cantora vocal vocalista **voz lírica feminina contrapõe** ferramenta implemento instrumento instrumento musical elegância atributos estilo maneira sabor categoria classe espécies género raça tipo comunicação mensagem **estilo** animação alma vida experiência intenção motivação motivo necessidade alguém indivíduo pessoa ser humano existência forma de vida etapa período tempo **vida** agrupação conjunto grupo homem homo sapiens humana humanidade humano ser humano alguém indivíduo pessoa amado amante namorada macho marido varão colega cónjuge cônjuge consorte esposa esposo adulto de maior terceira idade cara-metade parente **homem** esquema modelo afiguração desenho ficheiro gráfico figura ilustração imagem quadro exemplo exemplar adjunto ajudante assistente criado servente servidor **modelo reservado** classe classe social estrato alguém indivíduo pessoa ser humano mulher garota moça rapariga senhorita amado amante namorada colega cónjuge cônjuge consorte esposa esposo homem marido adulto de maior terceira idade fêmea **mulher**"
Hyponym	"portar exalar possuir ter estar formulado ou redigido com certas palavras constar gabar-se vestir **apresenta** ruído uníssono voz barítono baixo contralto alto mezzosoprano mezzo soprano mezzo-soprano mezzo-sopranos soprano tenor tenores **voz lírica feminina contrapõe** estilo maneira estilo de vida forma jeito barroco classicismo neoclassicismo romantismo tratamento moda alegoria dispositivo eloquência eloqüência grandiosidade jargão variação linguística fala estilos musicais genero musical género musical gênero musical poesia retórica verbosidade género literário gêneros literários **estilo** fauna fauna selvagem vida animal vida selvagem alacridade furor energia vigor biologia sobrevivência além além-túmulo outro mundo submundo ultravida vida após a morte dia-a-dia dias **vida** homo erectus homo habilis homo sapiens homem de neandertal espécie humana geração homem humanidade mundo povo garoto macho menino moço rapaz varão criança infante marido celibatário solteiro baixinho catatau garotinho guri mancebo mocinho petitinho petiz piquiticu rapazinho rapazote namorado noivo almofadinha dândi figurino galã janota eunoco eunuco gajo cavalheiro senhor cara tio avô velho patriarca branco viúvo bígamo consorte cônjuge esposo quem está na lua de mel recém-casado esposa mulher **homem** bola globo globo terrestre modelo planetário planetários molde norma técnica padrões ritmo supermodelo **modelo reservado** garina garota garotinha guria jovem menina menininha meninota moça mocinha moçoila puela mulher gata senhorita donzela funcionário operário princesa dona de casa marquesa negro dama de honra dama madame senhora gueixa amiga amante namorada heroína dona ménades mênades matriarca chefa directoria babá huri peripatética beleza celibatária solteira viúva viuvez esposa marido **mulher** "
Holonym	"**apresenta voz lírica feminina contrapõe estilo vida** casal casamento **homem modelo reservado mulher**"
GloVe	"apresentam apresentando possui mostra contém apresentou apresentar traz apresentava inclui apresentado **apresenta** vozes ouvir som cantar ouvido melodia vocal fala palavras guitarra tom **voz** poética camoniana **lírica** masculina feminino masculino campeã juvenil artística olímpica adulta **feminina** contrapôs **contrapõe** estilos inspirado gosto moderno barroco chammiya gótico tradicional típico inspiração original **estilo** viver sua história toda tempo vidas anos humana própria mesmo tudo **vida** rapaz mulher ele outro jovem alguém aquele indivíduo homens humano suspeito **homem** modelos conceito padrão novo protótipo sistema **modelo** reservada restrito **reservado** mãe esposa marido filha menina ela homem criança irmã jovem garota **mulher**"

```
1: procedure TopDown(State, Algorithms)
2:     New_States ← empty List
3:     for α ∈ Algorithms do
4:         New.members ← State.members ∪ α
5:         New.eval ← Evaluate(New.members)
6:         if New.eval > State.eval then
7:             New_States ← New_States ∪ New.
8:     return New_States.
```

Figure 5.2: *Pseudo-code for TopDown Algorithm*

augmented by Hyponym and, if it does not match words, it will call IR-E-Hyponym, which has the queries augmented, but the database is without any augmentation. The same applies to all similar relations. Word Embedding, similar to IR-H, uses the text as it is given, if it was augmented the algorithm uses it augmented.

5.4 Greedy Search

There is a commonsense in the field of AI that combining different algorithms that perform the same task in distinct ways leads to better performance compared to the algorithms alone. This is somehow supported by IBM's Watson's architecture [FBCC+10] having a massive amount of algorithms running in parallel and then combining them; Aristo combines different techniques through logistic regression of the answers of five different solvers [CEK+16]; as final example, the algorithms submitted to solve the Japanese Center Test also followed the same idea [FKKM14].

We have so far described a large number of algorithms, made of a base approach — Information Retrieval and Word Embedding — plus some variations: the database used, the augmentation method, etc. One approach for combining them is simply to assume that each method "votes" for a candidate-answer, and we select the most voted. However, using all algorithms introduces noise and hurts the performance; instead, we seek for a majority vote ensemble that uses only some of the algorithms. As we have available approximately 550 algorithms, we cannot compute all possible combinations to determine which one is the best. Instead we treat finding an ensemble as a greedy search problem using Beam Search. We define as state a selected group of algorithms.

We choose Beam Search because it can simulate different paths while maintaining an upper bound of states on memory. This technique requires two meta-parameters: an evaluation function — over a given state — and an integer known as beam width, that is, the maximum number of states that will be explored. In each iteration the algorithm generates the successors of the actual state for each state in the beam, then it evaluates each of these successors, sorts them and, finally, being β the size of the beam, selects the first β elements as new actual states. Our evaluation function — the same evaluation described in Section 6.1 — is based on the score that state achieves when solving the exam. To generate the successors of a state we have three functions: TopDown, BottomUp and Alteration, by definition the successors returned by each of these functions (if any) will always perform better than the state that generated them.

The first of the functions, TopDown, is responsible for generating the successors that are created by adding another algorithm to the actual selected algorithms. This is done by fixing the actual state and adding one more algorithm from the pool of (total) algorithms, evaluating this new combination and, if it improves the performance, saving it as a successor. Following, we replace the added algorithm by another of the pool and repeat the process until all the pool is tested. We present in Figure 5.2 a pseudo-code of this function.

The second function, BottomUp, looks for successors that use one less algorithm than the actual state. The intuition is that one algorithm may be added in the beginning, but some iterations latter it may be pulling down the group. This function takes one algorithm of the actual state and evaluates the remaining group, if it is better, then it is added to a list of successors. Next, the algorithm is

```
1: procedure BottomUp(State)
2:     New_States ← empty List
3:     for α ∈ State.members do
4:         New.members ← State.members \ α
5:         New.eval ← Evaluate(New.members)
6:         if New.eval > State.eval then
7:             New_States ← New_States ∪ New.
8:     return New_States.
```

Figure 5.3: *Pseudo-code for the BottomUp Algorithm*

```
1: procedure Alteration(State, Algorithms)
2:     New_States ← empty List
3:     for α ∈ State.members do
4:         Aux.members ← State.members \ α
5:         for β ∈ Algorithms do
6:             New.members ← Aux.members ∪ β
7:             New.eval ← Evaluate(New.members)
8:             if New.eval > State.eval then
9:                 New_States ← New_States ∪ New.
10:    return New_States.
```

Figure 5.4: *Pseudo-code for the Alteration Algorithm*

added back to state and other is taken, repeating the process until all algorithms were taken once. The pseudo-code for this function is showed in Figure 5.3.

The last function to generate the successors is the same as doing a BottomUp followed by TopDown: this function replaces the algorithms of the actual state for those of the pool. The intuition is that a algorithm α_1 may be good in a given combination but later on it would be better to have another one, say α_2, in the group instead of α_1. Figure 5.4 shows the pseudo-code for this function. The main idea is to replace each of the state's algorithms, one at a time, for each of pool's algorithm and evaluate to check if the ensemble becomes better.

Finally, the greedy search that combines the three previously described functions with Beam Search is presented in Figure 5.5. The idea is to have a list L of β nodes — each node is a state representing a different ensemble of algorithms and how well they perform on the exams —, removing all nodes as they are expanded using the functions TopDown, BottomUp and Alternation; having expanded the last node of L we will have a new list of unpredictable size, we sort this list by their performance on the exams, then we take the first β elements of the list and put them in L. The next step is to start again the process until there are no successors. Bearing in mind that each time a node is expanded we also verify if this node is better than the best seen so far, when L is empty we will have the best local ensemble, that is, the best ensemble reachable using the three functions and a beam of size β.

```
1: procedure SEARCH(Algorithms, Beam_width)
2:     State.members ← ∅
3:     State.eval ← 0
4:     L ← State
5:     while L ≠ ∅ do
6:         Aux ← ∅
7:         for s ∈ L do
8:             if s.eval > Best.eval then
9:                 Best ← s
10:            Aux ← Aux ∪ (TopDown(s, Algorithms) ∪ BottomUp(s) ∪ Alteration(s, Algorithms))
11:        Aux ← SortByEval(Aux)
12:        L ← Fetch(Aux, Beam_width)
13:    return Best
```

Figure 5.5: *Pseudo-code of the greedy search*

Chapter 6

Empirical Results

In this chapter we present the results achieved by the algorithms proposed in the previous chapter when solving questions from the ENEM dataset. In this part we created a "Preprocessed" version of the exams putting every word to lower case, removing stop-words and stripping punctuation. From this Preprocessed version we create the augmented ones. We selected only questions that do not require understanding image (IC), mathematical reasoning (MR) and treating Chemical Elements specially (CE), this sums up to 921 questions. The results of Information Retrieval and Word Embedding individually are presented in Section 6.2 and 6.3, respectively. Following we present in Section 6.4 the results of the ensemble and SVM. We start by defining how we evaluate the algorithms.

6.1 Evaluating the Algorithms

In order to evaluate the algorithms we use a metric based on the answer they output. As each candidate-answer (or alternative) receives a score, we select — always — the highest scoring candidate-answer. However, sometimes an indecision happens, in other words, multiple alternatives draw by receiving the same score, the highest. In order to avoid enforcing algorithms to chose randomly an answer in case of indecision, we penalize the algorithms dividing the correct answer reward by the length of the output answer.

We set 1 as the reward for giving the correct answer. Originally the ENEM sets different values for each question, depending on the number of students that answered it correctly, but these values are not publicly available. The algorithms receives 0 for a question if the (output) answer does not contain the correct answer. Consequently, the final evaluation is: if the answer does not contain the correct answer the evaluation is 0, if it contains, then being R the length of the output answer, the evaluation is $\frac{1}{R}$. This ensures that an algorithm giving constant score for every alternative receives the same (expected) evaluation as a random guesser.

For instance, being $Eval(\alpha, \gamma)$ the functions that receives two answers, the algorithm's answer (α) and the correct answer (γ), and returns the evaluation of that answer:

$$\begin{aligned} Eval(A, B) &= 0 \\ Eval(C, C) &= 1 \\ Eval(DE, D) &= \frac{1}{2} \\ Eval(ABC, D) &= 0 \\ Eval(ABCDE, E) &= \frac{1}{5} \end{aligned}$$

Then, the evaluation of an algorithm can be done by summing $Eval$ for every question, this sum equals the number of points it achieved. In this scenario, the accuracy of an algorithm can be

Table 6.1: *Results of Information Retrieval using the Header and its variations*

Variation	indecision	Total Points	Accuracy(%)	std
Normal	262	193.31	20.94	2.5
Preprocessed	317	195.90	21.23	3.0
Synonym	162	195.15	21.11	2.9
Hyperonym	130	222.33	24.13	3.4
Hyponym	122	208.81	22.63	4.2
Holonym	193	203.43	22.01	3.7
GloVe	113	201.43	21.80	4.1

defined as the number of points over the number of questions. This accuracy will also indicate the probability of that algorithm answering correctly a random question from the dataset.

Finally, we measure the accuracy of the algorithms in each exam and then take the accuracy's average and standard deviation (std), thus we evaluate the performance of the algorithms exam-wisely and how reliable it can be to an unseen (ENEM) exam.

6.2 Information Retrieval

Firstly we analyze the results achieved by the simplest of all heuristics, the **IR-H**, the one that uses only the text given as database to answer the question. We present in Table 6.1 the performance of the IR-H with its variations. In the third column we show the total number of points the variation achieved. In the fourth and fifth columns we present the accuracy of the variation exam-wisely and the standard deviation (std), respectively. This strategy has as characteristic frequently attributing zero to every alternative, we call this "indecision" and present in the second column the number of questions that had indecision.

The first point to notice in Table 6.1 is that the Normal variation is just a little better than a random guesser — achieving 20.9%, but its standard deviation (2.5) shows that in some exams this variation is worse than random. The second important point is that all augmentations[1] have higher accuracy than Normal, but their standard deviation is also higher. The number of indecision goes down with augmentations — except for Preprocessed, which takes words away. The augmentation that minimizes indecision is GloVe, which is the one that most adds words to the text. The best variation is Hyperonym; it halves the cases of indecision, it has the highest accuracy (24.13%) and most of the times is better than random.

As expected, the text given in the header do provide some information that can be used to answer the question. However, something else is necessary, here we only add words that have a relation with the ones appearing in the text and this (intellectually) shallow approach always increased the performance. The main disadvantage of this approach are: (i) the texts given are usually short, (ii) the questions do not ask for an ipsis litteris phrase of the text. To diminish these two drawbacks we try to solve an assessment using bigger databases; firstly using data given in the other exams, this is our **IR-E** strategy. As previously stated, this database uses the Normal variation of the exams, only the queries are augmented in the different variations. The results of this strategy can be seen in Table 6.2. This approach do not have cases of indecision, so the column is omitted.

The first point is that the accuracy of IR-E-Normal (23.37%) is better than the accuracy of IR-H-Normal (20.94%). This shows that using bigger knowledge bases may be advantageous and that the exams either have indeed limited domain or that it asks for recurrent themes. The second point is that the only augmentation that hurts the accuracy of this strategy is GloVe — the loss is smaller than 0.1%, but the standard deviation raised by 0.3. All variations are above random in average, but sometimes they perform worse than random, except for Hyperonym — 25% accuracy and 3.8 standard deviation, the lowest std of these variations. The third point is that although IR-E-Normal

[1] That is, every variation except for Normal.

Table 6.2: *Results of Information Retrieval ENEM and its variations*

Variation	Total Points	Accuracy(%)	Std
Normal	215.60	23.37	4.7
Preprocessed	218.05	23.67	4.5
Synonym	217.55	23.58	5.8
Hyperonym	229.95	25.02	3.8
Hyponym	229.13	24.90	5.1
Holonym	226.05	24.55	3.9
GloVe	214.88	23.33	5.0

Table 6.3: *Results of the variations of Information Retrieval using Wikipedia*

Variation	Total Points	Accuracy(%)	Std
Normal	247.75	26.90	3.8
Preprocessed	243.58	26.46	3.8
Synonym	219.50	23.88	3.4
Hyperonym	231.83	25.24	3.8
Hyponym	222.00	24.12	2.6
Holonym	228.08	24.81	3.7
GloVe	211.00	22.85	4.2

is 2.43% better than IR-H-Normal, IR-E-Hyperonym is only slightly better than IR-H-Hyperonym — 25% against 24.1%.

From this strategy we take that Hyperonym augmentation provides useful information to solve questions and also that a larger informative database is indeed useful. The next strategy, **IR-W**, uses an even larger database extracted from the Wikipedia. The results are presented in Table 6.3.

The main difference of this strategy to the previous ones is the best scoring variation: in this the best thing to do is not augmenting the queries. IR-W-Normal (26.9%) is better than IR-E-Normal (23.37%) and, consequently, IR-H-Normal (20.94%). By preprocessing the text the variation has a 0.5% drop in accuracy without any change in standard deviation (3.8). By augmenting using GloVe the accuracy drops by 4% and the deviation goes up to 4.2. On the other hand, by augmenting using the WordNet relations the standard deviation is at least as good as the Normal variation. Hyperonym is the WordNet's highest scoring variation (25.24%). Here happens something similar to what happened between IR-E and IR-H. That is, the fact that IR-W-Normal is better than IR-E-Normal by a relatively large margin — about 3% —, but the Hyperonym variation is just a bit better. In the actual case the margin is even smaller — 25.24% against 25.02%. Except for GloVe and Synonym, all the other variations are always better than random guessing.

From the performance of these three databases we can take that bigger databases have advantage and that the Hyperonym variation seems to be the most advantageous augmentation to use. However, either augmentation has a limit — for in IR-W it hurts the performance — or the way we are using it may be inserting too much words and thus adding too much noise. Which is made even worse by the fact that WordNet is not in Portuguese.

The following heuristics are attempts to combine the performances of the three previous strategies. The first attempt, **NDH**, calls either IR-E or IR-W when IR-H attributes score zero to every alternative, receiving the name of either NDH-E or NDH-W. The performance of this heuristic can be seen in Table 6.4.

It is possible to see that both NDH-E and NDH-W are always better than IR-H — when comparing the accuracy of the same variations, that is, the Normal variation of one with the Normal variation of the other and so on. By comparing NDH-E with IR-E we can see that most of the IR-E variations are better than NDH-E, the exceptions are NDH-E-Preprocessed and Hyperonym. The latter is the highest scoring (25.28%) variation of NDH-E; meaning that is advantageous to combine the two databases and augmentation. There are two variations of NDH-W that are better

Table 6.4: *Results of NDH-E and NDH-W*

NDH-E				NDH-W			
Variation	Points	Accuracy(%)	Std.	Variation	Points	Accuracy(%)	Std.
Normal	212.80	23.06	3.9	Normal	217.01	23.49	3.2
Preprocessed	218.73	23.71	4.4	Preprocessed	222.70	24.12	3.6
Synonym	209.28	22.69	3.3	Synonym	200.25	21.70	2.7
Hyperonym	233.06	25.28	4.1	Hyperonym	228.16	24.78	3.6
Hyponym	222.95	24.17	5.0	Hyponym	216.91	23.54	4.2
Holonym	217.46	23.51	4.4	Holonym	214.93	23.24	3.9
GloVe	212.36	22.98	4.8	GloVe	206.30	22.35	4.5

Table 6.5: *Results of Adding Heuristic*

AH-E				AH-W			
Variation	Points	Accuracy (%)	Std.	Variation	Points	Accuracy (%)	Std.
Normal	219.73	23.82	4.8	Normal	245.5	26.64	4.0
Preprocessed	216.73	23.52	4.8	Preprocessed	238.0	25.84	4.4
Synonym	221.23	23.97	5.6	Synonym	213.0	23.16	2.9
Hyperonym	226.93	24.72	3.6	Hyperonym	231.33	25.18	3.7
Hyponym	220.23	23.93	4.2	Hyponym	223.0	24.22	3.3
Holonym	221.23	23.95	4.3	Holonym	235.5	25.58	4.2
GloVe	215.73	23.41	5.7	GloVe	217.5	23.54	4.8

than IR-W, namely, Hyponym and Holonym. The best scoring variation of NDH-W is Hyperonym (24.78%), which is worse than IR-W-Hyperonym (25.24%).A second point to note is that NDH-W wins against NDH-E only in Normal and Preprocessed. The third point is that here, again, in both NDH-E and NDH-W, Hyperonym is the highest scoring variation; and between all variations of NDH — both NDH-E and NDH-W —, the highest scoring is NDH-E-Hyperonym (25.28% with 4.1 standard deviation). Additionally, taking standard deviation into account, most of the NDH variants have some performances worse than random.

The last attempt of the IR heuristics is the **AH**, which adds the scores — without any scaling — of IR-H with IR-E or IR-W. The results of AH-E and AH-W are presented in Table 6.5.

Similar to NDH, AH is always better than IR-H in accuracy, but the standard deviation of AH-E is at least as high as IR-H. The same follows to AH-W in comparison with IR-H, but AH-W has one case — Hyponym — which is lower than the respective IR-H variation. Four variations of IR-E win against AH-E, namely, Preprocessed, Hyperonym, Hyponym, Holonym. Three variations of AW-W win against IR-W; so, AW do not have a clear relation of accuracy superiority over IR-E and IR-W. Comparing AH with NDH we can see that AH-E and NDH-E does not have a clear winner. AH-W is always better than NDH-W in accuracy, on the other hand, AH-W always has higher standard deviation. All variations of AH are at least 3% above random guessing. But by analyzing the standard deviation we can see that only AH-E-Hyperonym (24.72% $\pm$ 3.6 std) remains above random in AH-E; and in AH-W only GloVe becomes below random. Comparing AH-E with AH-W we see that AH-W has the advantage: it loses in accuracy only in Synonym and in standard deviation only in Hyperonym (by 0.1). Finally, the best variation of AH-E is Hyperonym (24.72%) and AH-W's is Normal (26.64%).

We can notice that — except for IR-W and AH-W — in all IR strategies/heuristics the accuracy raised when using augmentation. Most of the time, in 5 out of 7 cases, Hyperonym was the best variation to use, in the other 2 cases the best thing to do was not augmenting the text. The highest scoring algorithm was IR-W-Normal (26.90% $\pm$ 3.8), followed by AH-W-Normal (26.64% $\pm$ 4.0), IR-W-Preprocessed (26.46% $\pm$ 3.8), AW-W-Preprocessed (25.84% $\pm$ 4.4), AH-W-Holonym (25.58% $\pm$ 4.2) and NDH-E-Hyperonym (25.28% $\pm$ 4.1). These results are far behind Aristo's IR solver, while

our best is 26.9% Aristo's is 60.6%. This might be due to: (1) difference in database size, ours is 1.3GB, Aristo's is 280GB; (2) they used 129 questions for testing, we used 921, they may not have enough questions to attest a truthful performance; (3) simply the questions' type and style, the ones used by Aristo are more "IR-friendly".

6.3 Word Embedding

Following, we present the results when solving the exams using Word Embedding. First we present Word2Vec (W2V), then Wang2Vec — hereafter Wang —, GloVe and, finally, FastText (Fast). We denote these four by *types*. Here we present only the accuracy and standard deviation (std) of each variation, meaning that we are omitting the "Points" and indecision columns, for the latter happens less than ten times.

This family of algorithms has the format *type-model-vector_size-variation-heuristic*. We compare the algorithms changing only one variable at time and we do not compare the types directly. By changing *variation* we analyze what is the best thing to do with the text when using vectors of a specific size of a specific model when using heuristic R1 or R2. When changing *heuristic* we are analyzing if when using Word Embedding the text or the statement is more important to solve a question, these two are also the points we add in this work. When changing the *model* and the *vector_size* where are changing the hyper-parameter of the learned Word Vector, these two were already set, we are just testing their performance on the task. We also compare the four combinations of model-heuristic in the same variation and same vector size.

We start our analysis with Word2Vec's performance presented in Table 6.6. **W2V-CBOW-50** has some interesting results: in R1 all variations are better than Normal, except one: GloVe. This might contribute to the idea that it does not make sense to augment a text using Word Embedding when the solver also uses it; here, for the first time, Synonym (25%±2.5) is the best variation, followed by Hyperonym (24.68%±2.1). The R2 results resemble IR-Wikipedia: the best variation is Normal (24.69%±3.1) and the worst is Hyperonym (22.64%±3.6). R2 beats R1 in 4 out of 7 variations, but the highest achieving is W2V-CBOW-50-Synonym-R1 (25%±2.5).

Analyzing **W2V-Skip-50** we see that the GloVe variation is worse than the others variations for both R1 and R2. Moreover, W2V-Skip-50-GloVe-R1 (20%) has the lowest accuracy presented in this work. In both R1 and R2 the Hyperonym (23.85% and 24.5%, respectively) variations were better than Normal (23.19% and 23.23%, respectively). Additionally, Hyperonym is the best variation of W2V-Skip-50-R2, while for R1 the best is Hyponym (25.14%). Skip-R2 beats Skip-R1 in 6 out 7 variations, losing only in Hyponym (R1: 25%, R2: 22%).

The second vector size available is 100. At first we can see that **W2V-CBOW-100** is an upgraded version of W2V-CBOW-50, its R1-Synonym (25.78%±3.7) and Hyperonym (25.78%±3.2) variations are as good as the top four of IR; the same can not be said about the relation of **W2V-Skip-100** with W2V-Skip-50: Skip-100 is most of the times better than 50 in the same variation, but is not always better. Again, the best W2V-CBOW-100-R2 variation is Normal (25.06%), while R1's is Synonym (25.78%). Between variations CBOW-100-R2 is always better than Skip-100-R1, and CBOW-100-R1 only loses in one variation to Skip-100-R1, and this defeat is only by 0.11%.

About **W2V-300**, W2V-CBOW-300-R1 is always better than W2V-CBOW-50-R1. CBOW-Hyperonym (25.23% and 26.16%) here are better than CBOW-Normal (23.86% and 22.51%) in both R1 and R2, the same happens in Skip. In fact, Skip-Hyperonym-R2 got 26.54%, the highest score so far in Word Embedding, only 0.36% behind IR-W-Normal. Skip-300-R1 is always better than W2V-Skip-R1 of inferior vector sizes, while Skip-300-R2 loses to Skip-50-R2 only in Holonym, by 0.37% and to Skip-100-R2 only in Normal by 1.56%.

The next is **W2V-600**. The CBOW model has Normal (R1: 22.95%, R2: 23.98%) as worst variation and Hyperonym (R1: 24.62%, R2: 27.13%) as best. In Skip model Hyperonym (R1: 25.12%, R2: 26.72%) is always better than Normal (R1: 24.71% , R2: 23.94%), the best variation of Skip-R1 is Holonym (25.26%) and of R2 is Hyperonym. CBOW-600-R1 only wins against CBOW-100-R1 in GloVe, by 0.13%. CBOW-600-R2 is always better than CBOW-300-R2 and its Hyperonym got

Table 6.6: *Results of W2V*

Word2Vec CBOW					
Variation	50	100	300	600	1000
Normal-R1	21.65 ± 3.8	23.99 ± 3.9	23.86 ± 3.3	22.95 ± 2.9	23.69 ± 4.2
Preprocessed-R1	23.12 ± 3.8	23.74 ± 3.6	23.74 ± 3.3	23.14 ± 2.9	22.73 ± 4.8
Synonym-R1	25.01 ± 2.5	25.78 ± 3.7	25.09 ± 3.6	23.82 ± 3.6	23.81 ± 3.3
Hyperonym-R1	24.68 ± 2.1	25.78 ± 3.2	25.23 ± 4.5	24.62 ± 4.1	24.85 ± 4.6
Hyponym-R1	24.22 ± 3.2	25.67 ± 3.2	25.24 ± 4.2	24.52 ± 4.5	24.51 ± 4.7
Holonym-R1	23.64 ± 2.8	24.16 ± 3.5	24.47 ± 3.7	23.31 ± 3.3	23.93 ± 3.9
GloVe-R1	20.86 ± 4.0	23.33 ± 3.9	23.34 ± 2.9	23.46 ± 2.5	22.70 ± 3.9
Normal-R2	24.69 ± 3.1	25.06 ± 2.2	22.51 ± 3.3	23.98 ± 4.6	20.97 ± 3.8
Preprocessed-R2	24.03 ± 4.1	24.01 ± 2.9	23.59 ± 3.8	24.46 ± 3.3	23.33 ± 3.8
Synonym-R2	22.88 ± 2.4	24.08 ± 3.0	25.06 ± 3.3	26.15 ± 3.5	25.02 ± 4.5
Hyperonym-R2	22.64 ± 3.6	23.48 ± 2.8	26.16 ± 3.4	27.13 ± 2.4	26.35 ± 4.4
Hyponym-R2	23.45 ± 3.9	24.76 ± 3.4	24.52 ± 3.4	25.18 ± 2.2	24.52 ± 3.9
Holonym-R2	24.55 ± 3.9	24.55 ± 3.6	23.29 ± 3.5	24.41 ± 3.7	23.92 ± 4.6
GloVe-R2	23.63 ± 3.1	23.91 ± 4.0	25.46 ± 2.4	25.91 ± 3.0	24.78 ± 3.1
Word2Vec SKIP					
	50	100	300	600	1000
Normal-R1	23.19 ± 4.6	23.43 ± 2.9	23.66 ± 3.5	24.71 ± 4.5	24.16 ± 4.2
Preprocessed-R1	22.51 ± 2.3	23.85 ± 3.9	25.29 ± 4.3	24.80 ± 4.3	23.52 ± 3.6
Synonym-R1	22.49 ± 3.2	23.74 ± 1.5	24.01 ± 2.8	23.47 ± 3.5	22.48 ± 2.7
Hyperonym-R1	23.85 ± 2.9	23.26 ± 3.3	25.02 ± 4.4	25.12 ± 5.3	24.46 ± 4.1
Hyponym-R1	25.14 ± 3.5	24.14 ± 3.5	25.33 ± 3.5	24.64 ± 3.8	23.76 ± 4.3
Holonym-R1	22.27 ± 4.0	23.46 ± 3.4	23.75 ± 4.6	25.26 ± 4.3	24.93 ± 3.6
GloVe-R1	20.00 ± 3.1	20.80 ± 2.8	23.82 ± 3.3	23.73 ± 3.8	23.41 ± 2.9
Normal-R2	23.23 ± 4.2	25.77 ± 3.3	24.21 ± 4.0	23.94 ± 3.3	24.74 ± 3.6
Preprocessed-R2	23.46 ± 4.2	24.36 ± 3.6	23.60 ± 4.3	24.23 ± 4.7	23.90 ± 4.8
Synonym-R2	23.11 ± 2.8	24.96 ± 3.6	26.27 ± 3.6	25.34 ± 3.9	24.35 ± 4.0
Hyperonym-R2	24.50 ± 3.5	24.41 ± 3.5	26.64 ± 5.5	26.72 ± 4.3	25.19 ± 3.9
Hyponym-R2	22.24 ± 2.3	23.34 ± 3.6	24.59 ± 3.8	24.88 ± 3.4	24.02 ± 3.5
Holonym-R2	24.00 ± 3.9	23.33 ± 3.8	23.63 ± 4.5	24.36 ± 5.3	23.49 ± 4.8
GloVe-R2	22.22 ± 2.6	21.66 ± 4.0	23.25 ± 3.9	23.88 ± 4.4	23.35 ± 4.8

27.13% accuracy, being the best algorithm so far. Skip-600-R2 always wins against Skip-50-R2, while Skip-600-R1 only loses in one against Skip-50-R1. Skip-600-R1 only loses one to CBOW-600-R1: Synonym, by 0.35%. While in R2 CBOW-600-R2 is always better than Skip-600-R2, and Skip-600-R2 is always better than CBOW-600-R1.

Finally, **W2V-1000**: CBOW-100-R1 is worse than CBOW-1000-R1 in 2 variations, and CBOW-300-R1 always wins against CBOW-1000-R1. Both R1 and R2 of W2V-Skip-1000 are worse than their respective version of Skip-600 — except for one case in both. Hypernym is always better than Normal, and in 3 out of 4 cases the Hyperonym variation is the highest scoring variation. GloVe, on the other hand, is the worst in Skip-R2 and CBOW-R1.

Analyzing these results, we can say that Hypernym was the variation that usually was the highest scoring. In 4 out of 5 in CBOW-R1 GloVe was worst than Normal, in CBOW-R2 was just in 2 out of 5; in Skip, augmenting by GloVe hurt the performance of the algorithm 9 out of 10 times. Hypernym improved CBOW's in 8 out of 10, and Skip in 8 out of 10. In 4 cases was possible to notice R2 outmatching a related R1 — while the opposite was not seen. About the vector size, 100 was better than other sizes in CBOW; in Skip, on the other hand, 300 and 600 were better. There were 3 cases were a CBOW won against a Skip, the contrary happened only once.

Following W2V we present Wang in Table 6.7. Starting with **Wang-CBOW-50**, Hyperonym (R1: 23.75%, R2:25.06%) here are better than Normal (R1: 21.79% ,R2: 24.89%) variation. Hyponym (24.82%) is the best in R1 and GloVe (20.23%) is the worst, R2 has Hyperonym as the best. R2 loses to R1 only in Hyponym by 0.65%. **Wang-Skip-50**'s worst variation is GloVe (R1: 21.13%, R2: 22.44%); Hyperonym is always better than Normal, in R2 the best one is Preprocessed (24.21%) and in R1 is Hyponym (23.78%). Wang-Skip-50-R2 is always better than Wang-Skip-50-R1, except for Hyponym by 0.76%. CBOW-50-R2 is better than Skip-50-R2 in every variation except by 1.17% in Preprocessed variation and it wins in every variation against Skip-R1.

Wang-100: CBOW has Hyperonym (R1: 24.65%, R2:24.43%) as better than Normal (R1:23.50%, R2:23.13%) and GloVe (R1:21.67%, R2: 22.52%) as worst. Different from W2V, Wang's CBOW-100 is not an upgrade from CBOW-50. In fact, CBOW-50-R2 wins in five variations against CBOW-100-R2, while in R1 the 100 version wins — except in Synonym, by 0.26%. Skip-R1 also has Hypernym (24.12%) as better than Normal (22.63%). CBOW-R2 is better than Skip-R1, except in one case and is always better than Skip-R2-100.

Next, **Wang-300**. The best from CBOW-R1 is Preprocessed (24.26%) and CBOW-R2's best is Normal (25.92%), both CBOWs have as worst GloVe (R1: 22.16%, R2: 22.47%). R2 beats R1 in all but one, Hyponym. Skip version: R1's highest scoring is Hyperonym (24.37%), R2's best is Synonym (24.83%). Skip-300-R1 wins against Skip-50-R1 except in one variation. Skip-300-R2 beats Skip-100-R2. CBOW-R2 wins but in two categories against Skip-300-R1.

Wang-600: CBOW's best are Hyponym (25.31%) for R1 and Holonym (24.56%) for R2, their worst is GloVe (R1: 22.89%, R2: 22.33%); in R2 Hyperonym (24.19%) is better than Normal (23.34%). CBOW-600-R1 is better than CBOW-50-R1 and loses only in one category to CBOW-300-R1. Skip: the highest scoring in R1 is Preprocessed (23.94%) and in R2 is Holonym (24.30%), but in both R1 and R2 Hyperonym (23.73% and 24.23%) is better than Normal (21.78% and 24.03%), R2's worst is GloVe (22.49%) and R1's is Normal (21.78%). Skip-600 beats Skip-50 in 7 out of 10 times, Skip-600-R2 loses only in one to Skip-100-R2. CBOW's R1 is better but in one than Skip-R1, CBOW's R2 also beats Skip-R1 but in one category.

Finally, **Wang-1000**. CBOW's best is Holonym (R1: 24.86%, R2: 24.58%) and worst is GloVe (R1: 22.67%, R2:22.12%); in R2 Hyperonym (24.20%) is better than Normal (23.44%), in R1 they are even (24.49%), differing in the standard deviation — Normal has 3.8 and Hyperonym has 5.1. CBOW-1000-R1 is better than CBOW-50-R1, loses in Preprocessed to 300. CBOW-1000-R2 loses in two to CBOW-100-R2 and CBOW-600-R2. Skip: R1's best is Hyperonym (24.47), R2's is Holonym (24.58%), the worst are Synonym (22.47%) for R1 and Hyponym (22.74%) for R2. Skip-1000-R2 wins against Skip-100-R2, beats 600 but in two; Skip-1000-R1 beats Skip-50-R1 and Skip-100-R1 but in one. CBOW-R1 loses only in one to Skip-R1, namely, in GloVe.

Differing from W2V, Wang's best variations are usually Holonym and Hyponym. Hyperonym

Table 6.7: *Results of Wang*

Wang2Vec CBOW					
Variation	50	100	300	600	1000
Normal	21.79 ± 2.9	23.50 ± 5.2	23.35 ± 3.8	24.76 ± 4.5	24.49 ± 3.8
Preprocessed	20.96 ± 3.5	22.02 ± 3.7	24.26 ± 3.1	23.24 ± 3.7	24.22 ± 3.4
Synonym	22.93 ± 2.8	22.67 ± 3.3	22.59 ± 4.3	24.48 ± 4.6	24.57 ± 4.4
Hyperonym	23.75 ± 3.4	24.65 ± 5.4	23.92 ± 6.0	23.97 ± 5.0	24.49 ± 5.1
Hyponym	24.82 ± 4.8	25.48 ± 3.9	24.14 ± 5.5	25.31 ± 6.1	24.86 ± 5.9
Holonym	21.51 ± 3.4	22.89 ± 4.2	23.36 ± 4.0	23.92 ± 4.4	24.87 ± 4.5
GloVe	20.23 ± 3.2	21.67 ± 4.9	22.16 ± 3.6	22.89 ± 5.3	22.67 ± 4.8
Normal-R2	24.89 ± 2.6	23.13 ± 4.0	25.92 ± 4.6	23.34 ± 3.3	23.44 ± 4.9
Preprocessed-R2	23.04 ± 2.3	23.11 ± 2.4	24.63 ± 2.9	23.68 ± 3.7	24.47 ± 4.4
Synonym-R2	24.70 ± 2.7	23.55 ± 3.0	24.26 ± 2.2	23.42 ± 3.1	23.76 ± 2.9
Hyperonym-R2	25.06 ± 3.0	24.43 ± 5.6	24.32 ± 4.0	24.19 ± 3.2	24.20 ± 3.2
Hyponym-R2	24.17 ± 3.9	23.13 ± 4.2	23.27 ± 4.6	23.75 ± 4.8	23.65 ± 4.5
Holonym-R2	23.95 ± 3.0	24.58 ± 4.4	24.44 ± 2.4	24.56 ± 5.0	24.58 ± 5.5
GloVe-R2	23.48 ± 4.8	22.52 ± 2.7	22.47 ± 3.5	22.33 ± 2.8	22.12 ± 2.5
Wang2Vec SKIP					
	50	100	300	600	1000
Normal-R1	21.48 ± 4.2	22.63 ± 4.7	22.31 ± 4.7	21.78 ± 5.3	22.92 ± 3.7
Preprocessed-R1	21.22 ± 3.2	22.15 ± 4.1	23.32 ± 5.2	23.94 ± 4.6	23.64 ± 3.5
Synonym-R1	21.60 ± 3.0	21.12 ± 4.0	21.60 ± 2.8	22.05 ± 4.2	22.47 ± 3.6
Hyperonym-R1	23.18 ± 5.4	24.12 ± 6.5	24.37 ± 5.9	23.73 ± 5.1	24.47 ± 5.0
Hyponym-R1	23.78 ± 4.7	24.24 ± 5.0	23.71 ± 4.6	23.73 ± 5.0	23.63 ± 4.3
Holonym-R1	21.81 ± 4.5	22.02 ± 4.2	23.25 ± 5.2	23.47 ± 5.2	23.19 ± 4.4
GloVe-R1	21.13 ± 3.8	22.15 ± 3.7	22.34 ± 3.3	22.04 ± 3.0	23.04 ± 2.9
Normal-R2	22.69 ± 3.1	22.46 ± 3.4	24.31 ± 4.1	24.03 ± 3.4	24.48 ± 5.1
Preprocessed-R2	24.21 ± 3.9	22.53 ± 4.3	23.99 ± 3.1	23.32 ± 3.7	24.41 ± 3.7
Synonym-R2	23.54 ± 1.9	22.87 ± 3.6	24.83 ± 4.0	23.73 ± 4.5	24.28 ± 3.6
Hyperonym-R2	23.55 ± 3.1	22.16 ± 4.2	23.59 ± 4.6	24.23 ± 5.6	23.30 ± 4.0
Hyponym-R2	23.02 ± 4.6	22.09 ± 5.6	23.42 ± 5.5	22.75 ± 5.6	22.74 ± 3.9
Holonym-R2	23.85 ± 3.4	23.45 ± 4.8	23.64 ± 4.1	24.30 ± 3.2	24.58 ± 2.8
GloVe-R2	22.44 ± 4.7	21.94 ± 3.4	23.67 ± 3.6	22.49 ± 4.0	23.90 ± 4.0

was better than Normal in 15 out of 20 cases and in 18 out of 20 GloVe hurt Wang's performance. R2 here also usually is better than R1, CBOW also usually performs better than Skip. Here, in CBOW-R2 the 100 vector size loses to 30 and 500, in R1 the 600-sized vector dominates the 300 and 50, while in Skip size 600 beats 100 and 50.

The third group, **GloVe**, does not have Skip and CBOW models, the only parameters that change are the hyper-parameter vector size and the heuristic. We present its results in Table 6.8. **GloVe-50**-R1 has Hyponym (22.92%) as best — following the pattern that Hyperonym (22.25%) is better than Normal (21.16%) — and Normal as worst. GloVe-50-R2 has as worst GloVe (21.61%) and best Preprocessed (22.82%). Then, the **100**-sized GloVe has also Normal (20.84%) as worst of R1, the best of R2 is also Preprocessed (24.13%). Hyperonym (R1: 22.68%, R2: 23.85%) here is also better than Normal (R1: 20.84%, R2: 22.63%) and the worst variation is GloVe (21.19%) for R1 and Synonym (22.20%) for R2. 100-R2 is at least as good as 50-R2 and only loses in Hyponym to 100-R1.

GloVe-300: the best is Preprocessed (23.51%) for R1 and Holonym (25.01%) for R2. Hyperonym (R1: 22.81%, R2: 24.45%) is better than Normal (R1: 21.92%, R2: 23.50%), the worst is Normal for R1 and Hyponym (22.43%) for R2. 300-R1 wins but in two against 50-R1, and wins in every variation against 100, 300-R2 also wins against 50-R2 and 100-R2 — losing in one against this

Table 6.8: *Results of GloVe*

GloVe R1					
Variation	50	100	300	600	1000
Normal-R1	21.16 ± 4.2	20.84 ± 4.0	21.92 ± 3.8	20.42 ± 3.2	21.12 ± 3.2
Preprocessed-R1	22.37 ± 4.1	22.20 ± 3.7	23.51 ± 3.4	22.90 ± 5.2	23.41 ± 4.8
Synonym-R1	21.88 ± 2.6	21.62 ± 3.5	21.71 ± 3.1	22.30 ± 3.0	21.98 ± 3.6
Hyperonym-R1	22.25 ± 6.6	22.68 ± 5.7	22.81 ± 4.4	22.83 ± 4.7	22.81 ± 5.3
Hyponym-R1	22.92 ± 3.7	22.72 ± 3.7	23.10 ± 4.1	23.54 ± 4.3	23.53 ± 4.8
Holonym-R1	21.38 ± 4.6	22.38 ± 3.3	23.19 ± 3.1	22.78 ± 4.0	22.77 ± 4.2
GloVe-R1	21.31 ± 4.2	21.19 ± 3.6	22.08 ± 3.0	22.08 ± 3.7	22.21 ± 3.9
GloVe R2					
	50	100	300	600	1000
Normal-R2	22.66 ± 4.3	22.63 ± 4.2	23.50 ± 3.4	23.82 ± 4.1	24.58 ± 3.8
Preprocessed-R2	22.82 ± 3.7	24.13 ± 3.5	24.97 ± 3.4	24.76 ± 3.6	24.22 ± 3.3
Synonym-R2	21.87 ± 3.7	22.20 ± 2.6	22.65 ± 3.2	22.45 ± 2.9	22.35 ± 2.9
Hyperonym-R2	21.91 ± 4.4	23.85 ± 4.5	24.45 ± 4.6	24.99 ± 4.2	24.79 ± 4.7
Hyponym-R2	21.71 ± 5.2	22.45 ± 4.6	22.43 ± 4.4	21.99 ± 4.7	22.54 ± 3.9
Holonym-R2	22.50 ± 2.4	23.79 ± 2.6	25.01 ± 3.8	24.82 ± 2.8	24.28 ± 3.1
GloVe-R2	21.61 ± 4.2	22.23 ± 3.5	22.45 ± 3.4	22.67 ± 3.7	22.57 ± 3.7

latter.

600-sized GloVe: in R1 the Normal (20.42%) variation is the worst and the best is Hyponym (23.54%); the best using R2 is Hyperonym (24.99%), the worst is Hyponym (21.99%). 600-R1 is better than 50-R1 except in Normal, the same is valid for 100. 600-R2 is better than 50 and 100 — except in Hyponym —. 600-R2 beats 600-R1 except in Hyponym. **GloVe 1000**: the worst for R1 is Normal (21.12%), for R2 is Synonym (22.35%); the best for R2 is Hyperonym (24.79%), for R1 is Hyponym (23.53%). 1000-R1 beats 50-R1 except in Normal, wins against 100, and is at least as good as 600. 1000-R2 beats 50-R2 and 100-R2.

GloVe using R1 seems to increase the performance with vector size, 50 is worse than 100, which is worse than 600 which worse than 1000. While when using R2 heuristic 300 and 1000 are better than 100 and 50. GloVe also displays R2 outmatching R1. One interesting point is that in R1 the GloVe variation is always better than Normal, but in R2 it always hurt the performance. The worst variation here is Normal, perhaps choosing GloVe to augment the text did not match with the others. Following the pattern of the previous approaches, in 9 out of 10 cases Hyperonym improved the performance in relation to Normal.

Finally, as last type of Word Embedding, we discuss the results of **FastText** as presented in Table 6.9, this group is the highest scoring so far. **CBOW-50** has Synonym (R1: 26.90%, R2: 24.62%) as best variation. Here also Hyperonym (R1: 26.17%, R2: 24.14%) is better than Normal (R1: 23.28%, R2: 22.77%), which is the worst in both. CBOW-R1 just loses in one to CBOW-R2, namely, Holonym. Skip: the best using R1 is Hyponym (25.24%), but Hyperonym (24.68%) is better than Normal (22.99%); the Normal variation (24.41%) is the best of CBOW-R2; the worst of R2 is GloVe (22.67%) and of R1 is Preprocessed (22.07%). CBOW-R1 wins against Skip-R1. **FastText-100**, CBOW's best using R1 is Hyponym (25.33%), having Preprocessed (22.46%) as worst; the highest scoring of R2 is Synonym (25.62%) and the lowest scoring is Hyponym (24.19%). In Skip, Synonym (22.77%) and Hyperonym (23.08%) are the worst variation for R1 and R2, respectively, and the best are Hyponym (25.01%) and Holonym (25.63%).

About **FastText-300**: CBOW's both versions have Hyperonym (R1: 26.35%, R2: 25.79%) as best, R2's worst is Normal (21.90%) and R1's is Preprocessed (23.03%). In Skip-R2 there is the best algorithm so far: FastText-Skip-300-Holonym-R2 with 27.23% accuracy, every algorithm of FastText-Skip-300-R2 is better than any other of Skip-300-R1, being Preprocessed and GloVe the worst of them (R1: 23.13%, R2: 25.38%, respectively); for R1 the best is Hyperonym (25.23%), 300

Table 6.9: *Results of FastText*

FastText CBOW					
Variation	50	100	300	600	1000
Normal	23.28 ± 2.3	23.81 ± 2.7	23.38 ± 3.0	23.52 ± 3.5	23.19 ± 3.3
Preprocessed	24.25 ± 3.2	22.46 ± 4.8	23.03 ± 4.1	24.01 ± 3.6	24.45 ± 3.5
Synonym	26.90 ± 5.0	25.14 ± 4.4	25.03 ± 3.9	25.30 ± 4.4	25.18 ± 4.2
Hyperonym	26.17 ± 5.3	25.06 ± 5.1	26.35 ± 5.3	25.83 ± 4.5	25.95 ± 5.3
Hyponym	25.57 ± 4.4	25.33 ± 4.9	25.37 ± 4.7	25.15 ± 3.6	25.47 ± 4.0
Holonym	23.39 ± 3.7	23.68 ± 5.1	24.18 ± 3.7	24.04 ± 4.3	24.12 ± 4.7
GloVe	23.92 ± 4.8	23.50 ± 4.1	23.04 ± 4.0	22.65 ± 2.6	23.52 ± 2.8
Normal-R2	22.77 ± 2.3	22.02 ± 2.6	21.90 ± 2.6	23.12 ± 2.6	22.23 ± 2.7
Preprocessed-R2	22.94 ± 4.8	24.21 ± 4.7	24.96 ± 3.8	24.74 ± 3.3	24.74 ± 3.8
Synonym-R2	24.62 ± 3.0	25.62 ± 3.3	25.38 ± 2.8	24.79 ± 2.0	25.03 ± 2.2
Hyperonym-R2	24.14 ± 4.1	25.19 ± 3.4	25.79 ± 2.6	25.90 ± 4.1	26.02 ± 4.0
Hyponym-R2	24.33 ± 4.6	24.19 ± 3.9	24.20 ± 3.7	24.19 ± 3.6	23.88 ± 3.9
Holonym-R2	23.55 ± 4.2	24.34 ± 4.6	24.78 ± 3.9	25.76 ± 3.9	25.53 ± 4.2
GloVe-R2	23.14 ± 4.9	24.60 ± 5.0	24.48 ± 4.2	23.51 ± 3.0	23.74 ± 3.8
FastText Skip					
	50	100	300	600	1000
Normal-R1	22.99 ± 3.8	23.20 ± 4.0	23.59 ± 4.3	22.03 ± 3.1	24.37 ± 4.9
Preprocessed-R1	22.07± 2.5	22.82 ± 2.1	23.13 ± 3.6	23.17 ± 3.9	24.43 ± 6.0
Synonym-R1	22.37 ± 3.4	22.77 ± 3.6	24.02 ± 3.4	22.56 ± 3.5	23.22 ± 3.8
Hyperonym-R1	24.68 ± 5.1	24.59 ± 4.5	25.23 ± 5.0	24.71 ± 5.0	25.49 ± 6.9
Hyponym-R1	25.24 ± 4.2	25.01 ± 3.3	24.81 ± 3.2	24.97 ± 4.3	25.22 ± 5.2
Holonym-R1	22.97 ± 3.3	23.55 ± 3.0	24.77 ± 4.2	23.97 ± 4.8	24.88 ± 6.1
GloVe-R1	23.27 ± 2.5	23.93 ± 3.9	23.47 ± 3.8	22.89 ± 4.6	23.21 ± 4.3
Normal-R2	24.41 ± 1.9	24.40 ± 3.6	26.48 ± 3.0	25.60 ± 5.2	27.13 ± 5.6
Preprocessed-R2	24.36 ± 3.4	25.46 ± 3.7	26.81 ± 3.9	26.04 ± 5.0	27.01 ± 5.3
Synonym-R2	24.22 ± 3.4	23.82 ± 2.1	26.57 ± 1.8	26.11 ± 1.4	26.05 ± 2.2
Hyperonym-R2	23.38 ± 4.7	23.08 ± 3.8	26.57 ± 3.3	26.28 ± 3.4	26.37 ± 3.4
Hyponym-R2	24.34 ± 4.9	24.96 ± 3.8	25.37 ± 4.3	25.24 ± 4.0	24.76 ± 3.4
Holonym-R2	24.40 ± 2.2	25.63 ± 4.4	27.23 ± 2.8	27.40 ± 4.9	28.06 ± 4.5
GloVe-R2	22.67 ± 3.7	24.18 ± 2.5	25.38 ± 3.4	25.11 ± 3.9	23.74 ± 3.1

is better than 50 for both R1 and R2, and 300-R2 is also better than 100-R2. Skip-300-R2 is better than CBOW-300 R1 and R2. **600**: CBOW-600's best is Hyperonym (R1: 25.83%, R2: 25.90%), the worst for R1 is GloVe (22.65%), for R2, Normal (23.12%). CBOW-600-R2 beats 50 in almost every variation, except Hyponym. Skip: 600-R1's best is Hyponym (24.97%) — again, Hyperonym (24.71%) is also better than Normal (22.03%), which is the worst. 600-Holonym-R2 (27.40%) is the best using R2 and also the best so far, the worst variation of Skip-600-R2 is GloVe (25.11%). Skip-R1-600 is worse than Skip-R1-300 except in two case, 600-R2 is better than 50-R2, 100-R2 and only wins in Holonym when compared to 300-R2; besides that, Skip-600-R2 beats Skip-600-R1. CBOW-600-R1 beats Skip-600-R1 in all but one category and loses to Skip-600-R2 in every variation; Skip-600-R2 wins against CBOW-600-R2 also in every category.

Finally, **FastText-1000**, its CBOWs have as best variation the Hyperonym (R1: 25.95%, R2: 26.02%) and Normal (R1: 23.19%, R2: 22.23%) as worst. CBOW-1000-R1 loses to CBOW-50-R1 except in two variations and wins against CBOW-100-R1, except in Normal. Skip: the best and worst of 1000-R1 are Hyperonym (25.49%) and GloVe (23.21%), the best of R2 is Holonym — the best scoring individual algorithm of this paper, scoring 28.06% — and the worst of R2 is GloVe (23.74%). Skip-R1-1000 is better than Skip-R1-50, losing only in two variations, when compared to 100-R1 it loses only in GloVe, additionally, it wins against 600-R1. Skip-1000-R2 wins against Skip-50-R2 and only loses to Skip-1000-R1 in one category, namely, Hyponym. Skip-R2 beats CBOW-R1 except in Hyponym and wins against CBOW-R2 in every category.

FastText is the highest scoring group of algorithms, having five combinations that achieved more than 27%, three of them coming from a Holonym augmentation. In eight cases the best variation was Hyperonym, while the worst was usually Normal. Three times R2 won against R1, but once happened the opposite, and six times Skip was better than CBOW, once happened the contrary. Regarding the vector size, 300 and 600 are better than 50 and 100, but in one case 600 lost to 300, the CBOW-1000 lost to CBOW-50, but in Skip this relation is inverted. Consequently, higher dimensions seem to have an advantage.

6.3.1 Analysis of Word Embedding

Regarding the performance of the two heuristics in the WE family: in general, R2 heuristic was better than R1 in terms of accuracy. This might suggest that, when the words already encapsulate some kind of meaning, the text given does not add relevant information and might even insert error in the algorithm. But we can not attest that this is the best approach to answer questions using Word Embedding, or that given better vectors we could solve perfectly the exams using this (these) heuristic(s).

Hyperonym was the best variation in two out of four types. In all types, most of the time, Hyperonym enhances the performance of the algorithm when compared to the Normal version — the same happened with the IR algorithms. In other type, Hyponym was the best, this suggests that knowledge of hierarchies is more valuable to solve questions than the other kinds of knowledge, as, for example, contextual co-occurrence (GloVe augmentation) or interchangeability of words (Synonym augmentation). GloVe in fact hurt the performance of W2V and Wang, half of the times it hurt FastText's and even hurt its own R2 heuristic, but improved its own R1. Although Hyperonym was the augmentation that most often increased the performance of Normal, Holonym was the one that led to the highest accuracy. In W2V and Wang the CBOW model was better than Skip, in FastText was the opposite. Regarding vector size, apparently 300 and 600 have an advantage compared to the others sizes: W2V-Skip, Wang-R1, GloVe's performance grew with vector size in R1 and FastText performed better with more than 300 dimensions.

In Table 6.10 we present the top 20 scoring algorithms along with their accuracy in percentage, the position of that accuracy in a decreasing rank of accuracy, the position of that algorithm in a rank of Average Mean Ranking (AMR) and, in the last column, the AMR of that algorithm. We tested 539 different algorithms, the best AMR is 130.6, the worst is 403.5. We can see that the algorithms in the top by accuracy are someway constant, because all algorithms in the top 20 accuracy are inside the top 40 of AMR. Another information we can take from the table is that

Table 6.10: *Top 20 scoring algorithms, their accuracy, standard deviation (std) average mean rank (AMR) and the position of this score in a rank (Pos-AMR).*

Name	Accuracy	Position-acc	Std	Pos–AMR	AMR
Fast-Skip-1000-Holo-R2	28.06	01	±4.5	04	148.9
Fast-Skip-600-Holo-R2	27.40	02	±4.9	07	153.6
Fast-Skip-300-Holo-R2	27.23	03	±2.8	01	130.6
W2V-CBOW-600-Hyper-R2	27.13	04	±2.4	02	144.5
Fast-Skip-1000-Normal-R2	27.13	05	±5.6	05	151.6
Fast-Skip-1000-Preprocessed-R2	27.01	06	±5.3	18	173.4
IR-W-Normal	26.90	07	±3.8	08	154.1
Fast-CBOW-50-Syn-R1	26.90	08	±5.0	14	168.4
Fast-Skip-300-Preprocessed-R2	26.81	09	±3.9	03	146.5
W2V-Skip-600-Hyper-R2	26.72	10	±4.3	02	144.5
W2V-Skip-300-Hyper-R2	26.64	11	±5.5	24	181.5
AH-W-Normal	26.64	12	±4.0	17	172.9
Fast-Skip-300-Syn-R2	26.57	13	±1.8	12	163.0
Fast-Skip-300-Hyper-R2	26.57	14	±3.3	09	155.2
Fast-Skip-300-Normal-R2	26.48	15	±3.0	16	170.5
IR-W-Preprocessed	26.46	16	±3.8	32	188.2
Fast-Skip-1000-Hyper-R2	26.37	17	±3.4	21	180.1
Fast-CBOW-300-Hyper-R1	26.35	18	±5.3	39	194.0
W2V-CBOW-1000-Hyper-R2	26.35	19	±4.4	34	189.5
Fast-Skip-600-Hyper-R2	26.28	20	±3.4	20	179.3

although CBOW seems to beat Skip, the best individual algorithms of Word Embedding are most of the time Skip using the R2 heuristic. IR ends up appearing 3 times, the three based on Wikipedia.

This concludes our presentation of the algorithms individually, in the next section we will explore a way to combine them and achieve better results.

6.4 Combining Algorithms

In the previous sections we presented and analyzed different algorithms to solve the exams, the best scoring algorithms ranged from 28% to 26%. In this section we present the results achieved when combining these individual algorithms. The first approach presented is a SVM trained to classify alternatives through features extracted using Word Embedding. The second is an ensemble found using greedy search.

6.4.1 SVM

The next solver uses the features extracted from Word Embedding presented in Section 5.2. Here we come across a point of difference between the exam used by Aristo and ours: what is the question in our exam? When using the algorithms individually we developed two different heuristics that deal exactly with this point: R1 heuristic treats the header and the statement as a question, R2, diversely, equals the statement to a question. In light of the results presented in the previous section, we chose to use R2 as heuristic to extract the features; Aristo used four different groups of W2V vectors trained in different corpora, we use the 35 vectors used previously. Consequently, we use 70 features per alternative, while Aristo uses eight. Finally, it is important to state that we use the Normal variation of the exams to extract these features — in principle we could use every variation or even all of them together. The latter would generate a large number of features and for the former we would not have a strong criteria to use one instead of other. Therefore we decided for using the original exam, that is, the one without augmentation.

Table 6.11: *Performance of the four kernels of SVM. Accuracy is presented in percentage.*

	Linear	Rbf	Poly	Sigmoid
Accuracy	26.67	23.61	25.31	20.83
Standard Deviation	3.2	4.3	3.0	6.2

We used the implementation made available by Scikit — the Python 3 library. We did an extra tenfold cross-validation to optimize the C parameter, keeping the others as default, except that we flagged that the base is not balanced. This results in the algorithm attributing different C values depending on the label. We did this because this algorithm is fed with features of an alternative, consequently there are 4 wrong examples for each right example. The weight of a label y in a batch (bin) is calculated by:

$$\text{weight}(y) = \frac{n\_samples}{n\_classes * bincount(y)}$$

We present in Table 6.11 the accuracy and standard deviation achieved by the four types of kernel available, namely, Linear, Rbf, Polynomial and Sigmoid. To answer a question we feed the features, the algorithm returns the confidence on that features expressing a true alternative and we select the alternative with the highest confidence.

The first thing to notice is that the four Kernels do better than random guessing in average, Sigmoid (20.83%) is almost equal to random even in average. Taking standard deviation into account, only Linear (26.67% ± 3.2) and Polynomial (25.31% ± 3.0) remain better than random exam-wisely.

If we consider only the Normal variations of all the previous algorithms, the Linear SVM solver would be the third highest achieving, losing to FastText-Skip-1000-Normal-R2 (27.13% ± 5.6) and IR-W-Normal (26.9% ± 3.8). The advantage of this SVM solver is its relatively low standard deviation. Considering all the (almost 540) previous algorithms and the Table 6.10, the Linear SVM solver would be in the top 20, more specifically, the eleventh position.

Our results using this solver differ significantly from Aristo's. This might be due to the difference in questions and/or the fact they have a more limited scope, Science. They can train their vectors in (four different) domain-specific texts, ours was trained in a broad range and all vectors came from the same source.

6.4.2 Greedy Search

Following, we present the results of the Ensemble. The first thing we show is that the combination of algorithms found by the greedy search tends to perform better as the beam size increases. The same thing would be expected from the number of iteration until it halts. This subsection has two parts, the first shows some characteristics of the Ensemble, and the second shows its results.

Firstly, we use the greedy search over the whole dataset in order to investigate the behavior of this solver. In the right side of Figure 6.1 we present the graph showing the performance of the best combinations found when using beam size from 1 to 100. We stop at 100 because it takes 3 days to compute and in one week the 120 was not done.

As we can see, when beam size equals one the result is simply the best scoring algorithm. Because if you take it out the performance drops to 0, if you try to substitute it with any other the result will be worse and if you try to combine it with other through majority voting the result will always be worse. Using a beam of size 2 we already get a 297.75 (32%) scoring combination that took 12 iterations to be discovered — the graph showing the number of iterations done by each beam size can be seen in the left side of Figure 6.1. It keeps going until it reaches 336 points (36.48%) found after 33 iterations, the resulting Ensemble is composed by 30 algorithms, and three of them are repeated. We interpret this as a type of learning: it is giving more importance to an algorithm than the others, or, alternatively, it is learning a integer-weighted majority voting.

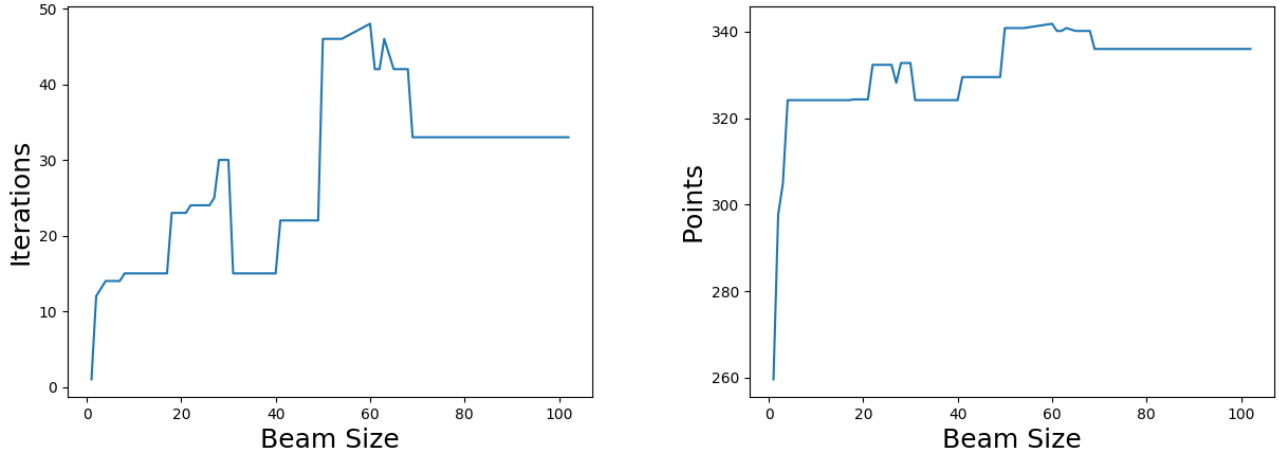


Figure 6.1: *The number of iterations (left) and points (right) done by the best combinations found when using a beam ranging from 1 to 100.*

We present the 30 selected algorithms in Table 6.12 along with their accuracy, rank in both accuracy and AMR (average mean rank); in the last column, ablation, is the accuracy of the Ensemble without that specific algorithm, in case of algorithm repetition — IR-W-Normal and Preprocessed — we present the successive ablation of that algorithm, that is, holding it off one time, two times and three times. By looking in the last column we can say that the sole algorithm that most influences the group is GloVe-100-Hyperonym-R2, which happens to be the third worst algorithm of the group in accuracy and AMR. On the other hand, the algorithm is most affected by holding all repetitions of IR-W-Normal, dropping its accuracy in 3.88%.

Moreover, we can see that the greedy search elected a broad range of algorithms: it took at least one of each variation (Normal, Synonym, etc.), 4 out of 7 families of IR — IR-E, IR-W and the two AH —, the four different types of Word Embedding, which 6 of them use R1 heuristic and 12 use R2, 6 are Skip and 12 CBOW. Patently, the repeated algorithms were all IR-based.

The next step is to analyze how similar these algorithms are. To do so we calculated the overlap of correct answers and mistakes of each algorithm with the others. We show in Table 6.13 the results for the five most significant algorithms selected, namely: IR-W-Normal, IR-W-Preprocessed, GloVe-100-Hypernym-R2, FastText-CBOW-600-Synonym-R1 and W2V-Skip-300-Hypernym-R2, which have their name simplified. In the second column we present the highest positive overlap, that is, how much an other algorithm agrees with it when it answers correctly a question; the third column is the average of positive overlaps an algorithm has with the others selected; following, in the fourth column we present the (highest) negative overlap. This is the case when the algorithm is wrong and the other algorithm chose the same wrong alternative; finally, the average of negative overlaps. The last line of the table presents the average of the averages of the group in positive and negative overlaps.

One thing to notice is that IR-W Normal and Preprocessed are very similar between themselves and both play an important role in this combination. As they are similar, their overlap is big, but their average overlap is not that big, meaning that although they agree between themselves, they tend to disagree with the others. The lowest overlap of IR-W-Normal is with FastText-CBOW-50-Synonym-R1, they agree positively with ratio 0.27 and negatively with ratio 0.19; Preprocessed has its lowest overlap with IR-E-Hyponym (0.28 positively) and FastText-CBOW-50-Synonym-R1 (0.19 negatively). Word2Vec's highest overlap comes from W2V-CBOW-600-Hypernym-R2 in both positive and negative overlaps. GloVe has the lowest highest overlaps of the five presented, both come from W2V-Skip-300-Hypernym-R2, its lowest positive overlap comes from AH-E-Holo, 0.28, and its negative comes from W2V-CBOW-1000-normal-R1 (0.23). Fast-600 has the largest highest overlap of the whole group, it comes from FastText-CBOW-1000-Synonym-R1, its lowest positive

Table 6.12: *Composition of the Ensemble along with their accuracy, average mean rank and its contribution to the group.*

Name	Acc.	Pos. Acc.	Pos. AMR	Ablation
IR-W-Hyponym	0.2412	222	182	0.3624
IR-W-Normal	0.2690	07	08	0.3567
IR-W-Normal	0.2690	07	08	0.3403
IR-W-Normal	0.2690	07	08	0.3260
FastText-Skip-1000-Preprocessed-R2	0.2701	06	18	0.3572
W2V-CBOW-300-Hyponym-R1	0.2524	65	88	0.3602
FastText-CBOW-1000-Synonym-R1	0.2518	73	67	0.3554
FastText-CBOW-600-Holonym-R2	0.2576	39	44	0.3555
IR-E-Hyponym	0.2490	101	133	0.3586
AH-E-Holonym	0.2395	242	148	0.3603
W2V-Skip-300-Hyperonym-R2	0.2664	11	24	0.3552
Wang-CBOW-1000-Holonym-R2	0.2458	143	137	0.3586
W2V-CBOW-1000-Normal-R1	0.2369	288	366	0.3565
AH-W-Hyponym	0.2422	199	158	0.3619
IR-E-Holonym	0.2455	150	140	0.3598
FastText-CBOW-50-Synonym-R1	0.2690	08	14	0.3554
W2V-CBOW-50-GloVe-R2	0.2363	298	357	0.3593
FastText-Skip-1000-Normal-R2	0.2713	05	05	0.3575
FastText-CBOW-1000-Hyponym-R1	0.2547	49	115	0.3608
IR-W-Preprocessed	0.2646	16	32	0.3572
IR-W-Preprocessed	0.2646	16	32	0.3395
AH-E-Hyperonym	0.2472	127	162	0.3602
GloVe-100-Hyperonym-R2	0.2385	261	312	0.3541
W2V-Skip-300-Synonym-R2	0.2627	21	06	0.3610
FastText-Skip-1000-Holonym-R2	0.2806	01	04	0.3564
W2V-Skip-100-Normal-R2	0.2577	38	56	0.3572
FastText-CBOW-600-Synonym-R1	0.2530	60	31	0.3546
FastText-CBOW-1000-Holonym-R2	0.2553	46	26	0.3569
Wang-CBOW-50-Holonym-R2	0.2395	241	181	0.3585
W2V-CBOW-600-Hyperonym-R2	0.2713	04	02	0.3566
Average:	0.2557	91.70	95.46	

Table 6.13: *Positive and Negative overlap of the five most influential algorithms of the Ensemble.*

Name	Highest P. Overlap	A. P. Overlap	Highest N. Overlap	A. N. Overlap
IR-W-Normal	0.9212	0.4389	0.8880	0.2982
W2V-300-Hyper	0.7669	0.4516	0.7314	0.3876
IR-W-Pre	0.9370	0.3867	0.8880	0.2901
GloVe-100-Hyper	0.6539	0.3993	0.5708	0.3345
Fast-600-Syn	0.9743	0.4221	0.9532	0.3629
Ensemble:	—	0.4129	—	0.3415

Table 6.14: *Performance of the Ensemble presented in percentage.*

year	train	test	size	iterations
2009	36.95	26.96	28(26)	32
2010	35.68	31.37	18(17)	19
2011	35.13	33.85	14(14)	15
2012	35.13	31.34	19(18)	21
2013	34.65	30.33	20(19)	21
2014	36.21	27.01	23(22)	25
2015	33.65	29.96	09(09)	10
2016	36.02	28.42	23(22)	27
2016(2)	36.00	27.95	14(14)	14
2017	33.96	26.68	09(09)	09
Avg:	35.33	29.39	17.69	—
Std. Dev.	0.98	2.25	5.9	—

overlap is 0.29, with Wang-CBOW-50-Holonym-R2, the negative is 0.21, with IR-W-Preprocessed.

When we analyze the whole group we can see that its positive overlap ratio is higher than the negative. This means that, when an algorithm is right, there are more algorithms that agree than when it is wrong. The problems with this are: the algorithms are more frequently wrong than right, and in case of being right, in average, the others will agree (overlap) with it with a ratio 0.4; meaning that in 60% of the time they will disagree even when the algorithm is right. The algorithms with highest positive average are: W2V-Skip-300-Hyperonym-R2 with 0.4516 and Wang-CBOW-1000-Holonym-R2 with 0.4504. The negative overlap, on the other hand should be the lowest possible, that means that when an algorithm is wrong the others will not agree with it. The lowest mean negative ratio is 0.29: IR-E-Holonym, IR-W-Normal, AH-E-Holonym, W2V-CBOW-1000-Normal-R1 and IR-W-Preprocessed. This may be the reason why IR-W-Normal and IR-W-Preprocessed were selected more than once: they have a relatively good positive overlap — 0.43 when the best is 0.45 — and some of the lowest negative overlap. They agree frequently, but they agree less frequently when they are wrong, and this is one of the traits of the selected group: the highest positive overlap is always larger than the highest negative overlap.

Having these characteristics being presented, we use the greedy search with cross-validation to find an Ensemble to solve the exams. We use arbitrarily beam size equals 10, because optimizing this parameter would be too expensive. Its performance is presented in Table 6.14. The second column represents the accuracy in the training set, the third column the accuracy in the test set. The size column represents how many algorithms compose the final ensemble and, in brackets, how many different algorithms there are in it. The last column presents how many iterations were necessary to compose the group.

The Ensemble tracks quite good its performance in the test set through the training set. In six cases the method repeated at least one algorithm. In all cases the ensemble was not composed by simply aggregating more algorithms, meaning that the three functions that compose the greedy search are relevant — otherwise the number of iterations would equal the size. Compared to all the previous approaches, the Ensemble has the highest accuracy (29.3%), this puts the method on the top position in Table 6.10. Additionally, the method has the eleventh smallest standard deviation (2.2%).

These things presented, we conclude the present chapter. We showed the performance of Information Retrieval with three different database and two different heuristics; then we showed the performance of Word Embedding using four embedding types, then a SVM and a Ensemble solver. In the following chapter we overview the presented results, compare the highest scoring algorithm of each approach and discuss some questions of the exam.

Chapter 7

Discussion

We start this chapter by reviewing the results achieved by each approach proposed; then we compare the highest scoring algorithm of each approach — Information Retrieval, Word Embedding, SVM and Ensemble — using the accuracy of each knowledge tag; finally, we present the questions that every individual algorithm (Information Retrieval and Word Embedding) got wrong.

7.1 Review of the Approaches

The first approach used was based on **Information Retrieval**, we proposed three databases with growing size: (1) only the text given in the question, (2) questions taken from other ENEM exams; (3) Wikipedia articles. Even the first database achieved, averagely, better than random, but only 0.94% above the 20% of a random guesser. Using the second database the performance went up to 23.37% and the third achieved 26.9%. Even though in average they are better than random, the first and second database have 2.5 and 4.7, respectively, of standard deviation, showing that sometimes they have a performance under random; the third database has 3.8 of standard deviation, showing that it performed better than random in all exams. Additionally we proposed two heuristics to combine the first database with the other two: either by using them in sequence (NDH) or by summing the two confidences (AH). These heuristics improved over the first database, but the accuracy achieved did not surpass the accuracy of the other base used.

In Figure 7.1 we show an example of question successfully answered by IR-W. In Table 7.1 we present the scores given by IR-W and IR-E for each answer. The document that gave the winning score for IR-W is a document on transpiration, containing almost all words of the query. IR-E has as best scoring document a text on an enzyme from the digestive system responsible for digestion — which has high overlap with the query. The word "sudorese" (sweating)[1] appearing in the query could be very discriminative, but it appears only one more time in the ENEM database, namely, as a symptom presented by a character in a poem. The top scoring document of the right alternative is on the difference of "heat" and "temperature" using boiling water as example. It is possible to say that IR-E got this question wrong because it did not have the information needed in its database.

Similarly, in Figure 7.2 and Table 7.2 we present a question that was answered correctly by IR-E. The top scoring document is the question 2009-106, a question on social and economical transformations caused by the "information explosion". The top scoring document retrieved by IR-W for candidate-answer A is a document about Marshall McLuhan, a philosopher of information theory, which has some connection with the answer, but it does not overlap enough to make it the first choice. The top scoring answer of IR-W is on globalization, in this document only "dissociated" and "countryside" do not appear in it. In fact, the word dissociated is what makes the alternative wrong, all the documents retrieved to this question have high overlap with the query, being the semantics of the candidate-answer very important to discriminate the correct answer — the top scoring algorithm of Word Embedding answered this question correctly.

[1] "Sudorese" means sweat, but the frequent word for it is "suor".

Tags
EK
Header
[...]
Statement
The main responsible for the maintenance of human body temperature is the
Answers
A- digestive system, because it produces enzymes that degrade high-calorie foods. B- immune system, because its cells act on blood, lowering heat conduction. C- nervous system, because it produces sweat, which allows for heat loss through water evaporation. [correct] D- reproductive system, because it secrets hormones that change temperature, mainly after menopause. E- endocrine system, because it makes antibodies that act on diameter variation of peripheral blood vessels.

Figure 7.1: *Question 11 of the 2009 exam. Header was suppressed.*

Table 7.1: *Scores given by IR-W and IR-E to Question 2009-11. The answer chosen is given in bold characters.*

Answer	Score IR-W	Score IR-E
A	42.71	**24.15**
B	44.54	17.17
C	**59.42**	21.42
D	41.63	16.03
E	36.64	17.26

Tags
EK TC
Header
[...]
Statement
Regarding the text and the cultural impacts caused by the diffusion of information technologies in the context of globalization, it follows that
Answers
A- the wide diffusion of information technologies in the urban centers and countryside increases the contact of different cultures, and at the same time it brings the necessity of reformulating the tradicional concepts of education. [correct] B- the appropriation of values and ideas from other cultures by a social group to favor itself is a source of conflicts and grudges. C- the social and cultural changes that come with the process of globalization reflect the preponderance of urban culture as well as make obsolete the traditional forms of education of the countryside. D- the populations in the big urban centers and in the countryside use the instruments and information technologies basically as means of mutual communication, and not as sources of education and culture. E- the intensification of the communication flow through electronic devices, a characteristic of globalization, is dissociated from the social and cultural development that occurs in the countryside.

Figure 7.2: *Question 72 of the 2009 exam. Header was suppressed because this strategy does not use it.*

Table 7.2: *Scores given by IR-W and IR-E to Question 2009-72. The answer chosen is given in bold characters.*

Answer	Score IR-W	Score IR-E
A	59.86	**34.84**
B	46.04	28.79
C	60.92	33.77
D	61.16	32.19
E	**68.25**	32.62

Table 7.3: *Number of times that the each variation was the highest and lowest scoring and number of times they improved the accuracy and standard deviation of the Information Retrieval approach.*

Variation	Raises Acc	Lowers Acc	Lowers Std.	Raises Std.	Highest	Lowest
Normal	—	—	—	—	2	1
Pre	4	3	2	4	0	0
Syn	3	4	5	2	0	2
Hyper	5	2	3	4	5	0
Hypo	5	2	3	4	0	0
Holo	4	3	3	4	0	0
GloVe	1	6	0	7	0	4

Moreover, we used six types of text augmentation, namely: Preprocessed, Synonym, Hyperonym, Hyponym, Holonym and GloVe. In general, if the the augmentation improved accuracy, it increased also the standard deviation; alternatively, if the augmentation worsened the accuracy, it lowered the standard deviation. Most of the times Hyperonym was the highest scoring augmentation, and in NDH-E the Hyperonym (25.28%) variation outperformed the usage of the two databases individually and with augmentation. In spite of that, the highest achieving algorithm was using the Wikipedia database without text augmentation. In Table 7.3 we show the number of times each variation improved or worsened the accuracy and the standard deviation of the Information Retrieval approaches — the number of attempts were seven — if the standard deviation did not increase we treat it as a improvement. Additionally, we point out in the table how many times a variation was the highest and lowest achiever in accuracy. It is possible to see the dominance of Hyperonym over the others — together with Hyponym — it was the one that raised the accuracy most of the times (5) and also was the highest scoring five out of seven times, without ever being the lowest; GloVe on the other hand most of the times (6) hurt the performance of the algorithms.

In Figure 7.3 we present question 2014-12. This question was not solvable by IR-H-Normal, but it was solvable by IR-H-Hyperonyms, as shown in Table 7.4. The Normal variation attributed zero score to the right answer due to having zero overlap of the candidate-answer with the text given. By augmenting the text using Hyperonyms a overlap was created — by connecting, for instance, "people" from the candidate-answer with "human" in the text — and became higher than the others, being able to answer a question that originally was tagged as requiring encyclopedic knowledge.

The second approach presented was based on **Word Embedding**, we used four different types

Table 7.4: *Scores given by IR-H-Normal and IR-H-Hyperonyms to Question 2014-12. The answer chosen is given in bold characters.*

Answer	Score IR-H-Normal	Score IR-H-Hyperonyms
A	0.00	**17.06**
B	1.61	5.90
C	**2.11**	7.48
D	1.61	7.95
E	2.03	5.88

Tags
EK
Header
Panayiotis Zavos “broke” human cloning last taboo — he transferred embryos to uterus that would manage them. This procedure is a crime in many countries. Apparently, the physician had a secret laboratory, in which he did his experiments. “I have no doubts that a cloned child will appear soon. I may not be the physician that will create him, but he will appear”, said Zavos. “If we dedicate ourselves, we can have a cloned baby in a year, or two, but I do not know if it is the case. We are not pressured to deliver a cloned baby to the world. We are pressured to deliver a healthy cloned baby to the world”.
Statement
Human cloning is a important topic in bioethics, a field that, among other things,
Answers
A- considers about the relation between the knowledge of life and people’s ethical values. [correct] B- legitimates the superiority of humankind above all other species on the planet. C- relativizes, in the case of human cloning, the usage of values such as right and wrong, good and evil. D- legalizes, through cloning techniques, the processes of human and animal cloning. E- substantiate technically and economically researches on stem cells to use in human beings.

Figure 7.3: *Question 12 of the 2014 exam. References suppressed.*

Table 7.5: *Scores given by IR-H-Normal and IR-H-Hyperonyms to Question 2011-82. The answer chosen is given in bold characters.*

Answer	Score FastText R2	Score FastText R1	IR-H-Holonyms
A	0.58	0.77	2.0
B	**0.62**	0.75	1.1
C	0.59	0.69	1.3
D	0.57	0.72	1.8
E	0.61	**0.82**	**2.3**

— Word2Vec, Wang2Vec, GloVe and FastText. Each of them having vectors of size 50, 100, 300, 600 and 1000; three types have both CBOW and Skip models and we have two different heuristics (R1 and R2) to use with each embedding. In general, the R2 heuristic was better than R1. The two bigger sizes of vector usually had advantage over the others. More difficult to analyze is the difference in the models: CBOW usually has better performance than Skip, but most of the top 20 scoring algorithms are using the Skip model. This means that most of the CBOW variations are better than the Skip variations, but, when Skip wins, this one variation scores higher than its related algorithms. All algorithms of this approach scored, in average, better than random.

In Figure 7.4 and Table 7.5 we present question 2011-82, a question that requires some inference on the text given. To this question the R2 heuristic — that is, ignoring the text — was better than R1. This might be due to the fact that while there is a big overlap of words from the candidate-answer and the text, incineration and chimney filters have some contextual co-occurrence. In fact, IR-H-Holonyms chooses answer E (score 2.3) and the right answer has the lowest score (1.1), showing that the big text overlap in this case is deceiving to the heuristic.

In Figure 7.5 and Image 7.6 we present question 2012-46, a question that requires understanding the passage given in the text and relating it to some concepts. In this question R1 was better than R2, as the text is important and there is no (expected) usual relation between the words “energy conversion”, “toy car” and “sling”. As shown by the scores of IH-H-Holonyms, there were no overlapping words in this question.

Most of the Word2Vec Normal variations have some performances below random guessing, that is pointed out by their standard deviation, such as Word2Vec-CBOW-1000-Normal-R1 (23.6 ±

Tags
EK
Header
One of the processes used in waste treatment is incineration, which has advantages and disadvantages. In São Paulo, for instance, the waste is burned in high temperatures and part of the released energy is transformed into electric energy. However, the incineration emits pollutants to the atmosphere.
Statement
One alternative to minimize the disadvantage of incineration, highlighted in the text, is to
Answers
A- increase the incinerated waste volume in order to increase the production of electric energy. B- encourage the usage of filters in the incinerator's chimneys in order to diminish the air pollution. [correct] C- increase the waste volume to decrease the operational costs related to the process. D- encourage the recycling waste collection in the cities to increase the incinerated waste volume. E- decrease the waste incineration temperature to produce more electric energy.

Figure 7.4: *Question 82 of the 2011 exam.*

Tags
RT DS
Header
Toy cars can be of different types. Among them, there are the ones that are winded up, a spring is compressed when the child pulls the car backwards. When it is released, the toy car begins to move while its spring returns to its original form.
Statement
The energy conversion process that occurs in the toy car described also occurs in
Answers
A- dynamos. B- automobile breaks. C- combustion engines. D- hydroelectric plants. E- slings. [correct]

Figure 7.5: *Question 46 of the 2012 exam.*

Table 7.6: *Scores given by FastText-Skip-1000-Holonyms (R2 and R1) and IR-H-Holonyms to Question 2012-46. The answer chosen is given in bold characters.*

Answer	Score FastText R2	Score FastText R1	IR-H-Holonyms
A	0.18	0.21	**0.0**
B	0.35	0.46	**0.0**
C	**0.38**	0.43	**0.0**
D	0.34	0.31	**0.0**
E	0.28	**0.48**	**0.0**

Table 7.7: *Number of times that the each variation was the highest and lowest scoring and number of times they improved the accuracy and standard deviation of the Word2Vec algorithms.*

Variation	Raises Acc	Lowers Acc	Lowers Std.	Raises Std.	Highest	Lowest
Normal	—	—	—	—	3	5
Pre	10	10	10	10	0	0
Syn	11	9	13	7	2	2
Hyper	16	4	6	14	9	2
Hypo	12	8	9	11	4	0
Holo	13	7	8	12	2	0
GloVe	5	15	15	5	0	11

Table 7.8: *Number of times that the each variation was the highest and lowest scoring and number of times they improved the accuracy and standard deviation of the Wang2Vec algorithms.*

Variation	Raises Acc	Lowers Acc	Lowers Std.	Raises Std.	Highest	Lowest
Normal	—	—	—	—	1	1
Pre	7	13	15	5	3	1
Syn	10	10	12	8	2	2
Hyper	15	4	6	14	4	0
Hypo	13	6	5	15	5	2
Holo	13	7	8	12	5	0
GloVe	2	18	13	7	0	12

4.2) and Word2Vec-Skip-50-Normal-R1 (23.19 ± 4.6); the highest scoring algorithm of Word2Vec were: CBOW-600-Hyperonym-R2 (27.13% ± 2.4), Skip-600-Hyperonym-R2 (26.72% ± 4.3), Skip-300-Hyper-R2 (26.64% ± 5.5), CBOW-1000-Hyper-R2 (26.35% ± 4.4). The second lowest standard deviation of all the individual algorithms come from Skip-100-Syn-R1 (23.74% ± 1.5). In Table 7.7 we present the relation of the variations with the increase or decrease in performance of the Word2Vec type, along with the number of times a variation was the best and the worst — in case of draw all the drawing variations score. Hyperonym is the augmentation that most often (15 out of 20) increases the performance. GloVe is the one that most often (15 out of 20) decreases the performance of the algorithm. These two are also the most frequent highest and lowest scoring, respectively; we highlight the Synonym augmentation: it increases more frequently than lowers the accuracy (11-9) and decreases the standard deviation more frequently than increases (13-7). GloVe most of the times lowers the standard deviation (15 out of 20) and Hyperonym most of the times (14 out of 20) increases the standard deviation.

The second type presented was Wang2Vec, this type using R1 heuristic almost ever has a standard deviation that shows performance under the random guessing, while using R2 makes an above random performance more frequent. The three best scoring algorithms of this type appear in the 30^{th}, 48^{th} and 59^{th} positions in accuracy rank, they are, respectively: CBOW-300-Normal-R2 (25.92% ± 4.6), CBOW-100-Hyponym-R1 (25.48% ± 3.9), CBOW-600-Hyponym-R1 (25.31% ± 6.1). The algorithm with the lowest standard deviation of this type is Skip-50-Synonym-R2 (23.54% ± 1.9), the fourth lowest standard deviation of all algorithms. In Table 7.8 we show the relation of the variations in relation to the accuracy and standard deviation. Hyperonym, Hyponym and Holonym raise the accuracy more often than lower it. Hyponym and Holonym are the augmentation that most times achieved the highest accuracy (5 times each), but Hyponym two times was the lowest accuracy. Hyponym was the highest 4 times and never the lowest; as with Word2Vec, if the augmentation tends to raise the accuracy, it also tends to raise the standard deviation, the opposite is also true — videlicet GloVe, it lowered the accuracy 18 times and lowered the standard deviation 13 times. Hyperonym raised the accuracy 15 times and the standard deviation 14 times.

From the third type, GloVe, only eleven of the seventy algorithms had an above random performance in all exams. The highest scoring algorithm of GloVe was 300-Holonym-R2 (25.01% ±

Table 7.9: *Number of times that the each variation was the highest and lowest scoring and number of times they improved the accuracy and standard deviation of the GloVe algorithms.*

Variation	Raises Acc	Lowers Acc	Lowers Std.	Raises Std.	Highest	Lowest
Normal	—	—	—	—	0	4
Pre	9	1	8	2	3	0
Syn	4	6	9	1	0	3
Hyper	9	1	0	10	2	0
Hypo	5	5	2	8	4	2
Holo	8	2	6	4	1	0
GloVe	5	5	8	2	0	1

Table 7.10: *Number of times that the each variation was the highest and lowest scoring and number of times they improved the accuracy and standard deviation of the FastText algorithms.*

Variation	Raises Acc	Lowers Acc	Lowers Std.	Raises Std.	Highest	Lowest
Normal	—	—	—	—	1	8
Pre	13	7	5	15	0	4
Syn	15	6	10	10	3	1
Hyper	17	3	3	17	8	1
Hypo	16	4	4	16	4	1
Holo	17	3	5	15	4	0
GloVe	11	9	10	10	0	5

3.8), the 93^{th} individual algorithm. In the top 10 worst algorithms, there is one of Information Retrieval (IR-H-Normal) and there are three Wang2Vec, three Word2Vec and three GloVe. The lowest standard deviation of GloVe comes from 50-Holonym-R2 (22.50% ± 2.4), the 26^{th} lowest standard deviation of all the individual algorithms. In Table 7.9 we show the interaction of GloVe with the augmentations; we highlight Preprocessed and Hyperonym in this type, both increased the accuracy 9 out of 10 times; and differing from all the previous approaches, GloVe augmentation here draw in how many times it hurts and benefits the algorithm. In this type the Normal variation was often (4 out of 10) the worst variation, Hyponym was 4 times the highest scoring and 2 times the worst. Preprocessed and Hyperonym were 3 and 2 times, respectively, the highest scoring. Regarding the standard deviation, Synonym, Preprocessed and GloVe were the ones that most of the times lowered the deviation; showing that GloVe algorithms are the ones that most benefit from GloVe augmentation.

As the last type of Word Embedding, we presented the results of FastText, the most successful type of solvers. Thirteen out of the 20 highest scoring algorithms and 11 out of 20 lowest standard deviation are FastText. The highest accuracy comes from Skip-1000-Holonym-R2 (28.06% ± 4.5) and the lowest deviation from Skip-600-Synonym-R2 (26.11 ± 1.4). In Table 7.10 we present the relation of the variations of this type with accuracy and standard deviation. The characteristic of this type is that it benefits most of the times with the augmentations, even with GloVe. Hyperonym and Holonym the ones that most of the times (17 out of 20) increased the accuracy; Hyperonym was the highest scoring most of the times (8) and Holonym was 4 times, without ever being the worst. Finally, concerning the standard deviation of this type: as with the others, the augmentation increase the standard deviation, there was no augmentation that lowered the standard deviation more often than increased. However, Synonym and GloVe drawn by lowering 10 times and raising 10 times the standard deviation.

From these previous results we can say that a positive point of the **Text Augmentation** strategy is that it increases the accuracy of all algorithms, not only of Information Retrieval, but also of the Word Embedding related algorithms; a negative point is that it also tends to increase the standard deviation of the algorithms, making the performance less reliable exam-wisely. We can see text augmentation as the addition of extra knowledge to the algorithms. In general, the

Table 7.11: *Comparison of performance based on question tags. The highest scoring algorithms of each approach were selected. Scores are in percentage.*

	TC	EK	DS	TC_{only}	EK_{only}	DS_{only}
#Questions	778	411	176	402	98	18
IR	26.4	28.4	25.5	25.8	33.6	16.6
WE	28.7	27.4	28.4	29.8	23.4	27.7
SVM	26.9	25.5	23.8	28.6	27.5	16.6
Ensemble	32.5	32.3	28.9	33.5	37.7	16.6

augmentations based on WordNet were better than augmenting by GloVe. We expect that an originally Portuguese WordNet would benefit even more the algorithms; moreover, we added all the words returned by WordNet, this makes the texts much bigger. This, allied with the fact that this word ontology is not originally in Portuguese, may insert too much noise to the algorithms.

Following we used **SVM** with features extracted using all the Word Embedding presented previously. The best result was using a linear kernel, it achieved 26.67% accuracy with 3.2 of standard deviation. This result is competitive with the other algorithms, it would be the third highest scoring Normal variation, with the lowest standard deviation. Lastly, we created an **Ensemble** through a greedy search, this Ensemble achieved the highest accuracy of this work (29.3%) with a top 11 lowest standard deviation (2.2). One characteristic of this solver is that it is based on overlapping correct answers — when an algorithm is right it is expected to have others agreeing with it —, but the algorithms are right about 23% of the time; increasing the overall performance of the individual algorithms will probably contribute to this strategy.

7.2 Accuracy per Knowledge Tag

In this section we compare algorithms based on their performance on the knowledge tags, videlicet: Text Comprehension (TC), Encyclopedic Knowledge (EK) and Domain Specific (DS). We treat the whole dataset as one exam, without differentiating between years.

In Table 7.11 we present the performance per knowledge tag of the highest scoring algorithms of Information Retrieval, Word Embedding, SVM and Ensemble — namely: IR-W-Normal (26.9%), FastText-Skip-1000-Holonym-R2 (28%), SVM-Linear (26.6%) and Ensemble (29%). We present the performance of the algorithms in the "subgroup" of pure questions, that is, the questions with only one knowledge tags. So "TC_{only}" is a subgroup of "TC". Importantly, note that there are only 18 questions tagged solely as Domain Specific, so the estimation of the accuracy in this partition is probably imprecise.

As expected for Information Retrieval, its best performance is on questions that rely solely on EK (33.6%) and its worst performance is on questions that rely exclusively on DS (16.6%). In this last case the algorithm performs worse than random. Word Embedding also fulfills the expected: its better performance is in purely TC questions (29.8%), and in all tags its performance is better than random. WE is the only algorithm — of the four presented in this table — that is better than random in exclusive DS questions (27.7%). SVM seems like a weaker version of WE, but its performance in questions of pure EK (27.5%) is better that WE. The last, the Ensemble, has the best performance in all tags, except for DS_{only}, achieving about 30% in all except for one tag; its best performance is in exclusive EK (37.7%) and the worst is in exclusive DS (16%).

From this last table we can take that: (i) IR and WE are complementary in its strengths; (ii) SVM without augmentation is competitive with the other approaches; (iii) the Ensemble does not equal, but surpasses, the best performance of its components; (iv) DS remains a challenge.

Table 7.12: *Questions that every algorithm mistook*

Year	Colors	Questions
2009	B,B	7, 59, 82, 113, 122
2010	W,Y	86
2011	W,Y	52, 71, 116
2012	W,Y	30, 65, 79, 101
2013	B,Y	27, 51
2014	B,P	60, 99
2015	B,P	25, 112
2016	Y,Y	107
2016(2)	Y,Y	1, 2, 62, 69, 78
2017	Y,Y	11, 97

7.3 Analysis of Questions

In this section we present the questions that no individual algorithm was able to answer correctly. There are in total 27 instances of such questions spread over the ten exams; these instances are presented in Table 7.12 — the Colors column presents the color of the exam[2], "B" stands for blue, "W" for white, "Y" for yellow and "P" for pink.

It is possible to classify these 27 questions in 6 different groups:

1. Specific concept; [9 questions]
2. Emotional thinking; [3 questions];
3. Text genre; [3 questions]
4. Conditioned thinking; [2 questions]
5. Simple flaws. [7 questions]
6. Others; [3 questions]

The first group is composed by questions which knowing a concept is the key point to answer the question correctly. They are in contrast with questions that ask for the definition of a concept. For instance, 2014-**60** required understanding competition in a biological context — this question is presented in Figure 7.6. 2016(2)-**62** required adaptation mechanisms of plants. One question required knowing how a centrifuge works (2017-**97**), other, 2016(2)-**78**, what is a Composting Facility. 2016(2)-**2** requires knowing pluriactivity. 2012-**65** requires understanding the reaction of two chemical elements, 2011-**52** is on electric current. The final two questions requires a combination of physics and chemistry: Green Energy (2013-**51**), and expansion and contraction of fluids (2016(2)-**69**).

We put in the **second group** questions that require identifying emotions conveyed in a piece of text. There are three questions in this group, namely: 2015-**112**, which required identifying humor, 2016-**107** asked for affection, 2017-**11** requires affection, longing, uneasiness and resentment.

The **third group** requires understanding text genre, that is, identifying to which genre a piece of text belongs to (2014-**99**); explaining why a certain style is used (2011-**116**), and in 2009-**122** it is necessary to justify the usage of the language's standard form[3] — this last question is presented in Figure 7.7.

The **fourth group** are questions that have a "conditioned thinking" aspect, that is, request what is the best in a given situation: what is the best type of energy to a given village (2010-**86**) and 2011-**71** inquires which sonar is the best to be used in a given situation.

[2]The exams have questions disposed in different patterns, these patterns are represented by different colors, usually there are four: Blue, Yellow, White and Pink.

[3]In Portuguese: forma culta.

Tags
DS TC
Header
There are bacteria that inhibit the growth of a fungus that by consuming the iron available in the environment causes diseases to the tomato plant. The bacteria also fixate nitrogen, make calcium available and produce auxins, substances that stimulate the growth of tomato plant.
Statement
Which of the biological processes below states an ecological relation of competition?
Alternatives
(a) Fixation of nitrogen to the tomato plant. (b) Making calcium available to the tomato plant. (c) Reduction in the quantity of iron available to the fungus. [correct] (d) Liberation of substances that inhibit the fungus' growth. (e) Liberation of auxins that stimulate the tomato plant's growth.

Figure 7.6: *Example of question which requires knowing a domain specific concept. References suppressed.*

Tags
TC
Header
When I talk to you I try to use your code. The image of the Indian in now-a-days Brazil cannot be the same of 500 years ago, from that past that represents that first contact. The same way Brazil now is not the same Brazil from yesterday, it has 160 million people with different surnames. Asians, Europeans and Africans came here and everybody wants to be Brazilian. The important question that we make is: which part of you is Indian? Your hair? Your eyes? The name of your street? You probably have an Indian part inside of you. To us, the important is that you see us as humans beings, as people that do not even need paternalism nor privileges. We do not want to take Brazil from you, we want to share it with you.
Statement
In the situation from which the text was taken, the standard form of the language is used to
Alternatives
(a) demonstrate the clarity and complexity of our language. (b) put both sides of the conversation in symmetric positions. [correct] (c) prove the importance of grammar in everyday conversation. (d) show how Indigenous languages were aggregated to the Portuguese language. (e) emphasize the importance of the linguistic code that we adopted as national language.

Figure 7.7: *Example of question which requires understanding types of text*

The **fifth group** is made by those questions that seem to be simple errors, mainly of Information Retrieval. They have the appearance of what should be correctly answered, but they are not, probably due to insufficient data. For example, one of these questions asked for Aristotle's definition of happiness (2013-**27**); some asked for characteristics of something, like 2012-**30** asked for the pattern of villages created in Amazon, 2016(2)-**1** asked for the main point of Greek tragedy as exemplified in King Oedipus; 2009-**113** demanded what is Digital Division. There is one question about the discoveries of a named scientist (2012-**79**). 2009-**7** required relating insulin to diabetes, a correlation that should be found easily. 2009-**59** required identifying which sentence summarizes better a text given.

The **last group** is composed by questions that do not have others similar, like 2012-**101**, which asks what a character in a poem concludes. 2015-**25** is on the relation between the two given texts; 2009-**82** seems like a question of General Knowledge or Actualities, it asks why exploring petrol in the North Pole is desirable, the answer is that the North Pole is more stable than Middle Orient.

Apart from the fifth group — Simple Flaws — the other groups are composed by questions which the algorithms used in this work are not supposed to perform well. Mainly the group 2 — for they require the methods to understand human emotions and/or behavior —; arguably there are researches in using Word Embedding to classify emotions [OZLL15], but the embedding are usually specialized to this task. None of the methods used are also expected to be able to categorize text as belonging to a certain type of genre nor to classify passages as having a certain kind of figure of speech (group 3). The groups one and four may be seen as errors of Word Embedding, since conditioned thinking may look like one example given, videlicet: $vector(\text{Deutschland}) + vector(\text{Capital}) \approx vector(\text{Berlin})$, and the vectors are expected to encapsulate some knowledge about the words, but they were not enough to answer correctly these questions, which might require a more sophisticated knowledge representation or usage.

We stress that although the total number of questions that all algorithms got wrong is small — about 3% of the questions —, this does not mean necessarily that it is possible to answer 97% of the dataset using only the approaches of this work. In fact, as some algorithms have performance under random-guessing exam-wisely, some questions might be answer correctly by guessing and are not identified here. Additionally, although 3% of the questions were not solved correctly by any algorithm, there was no question answered correctly by every algorithm. So, more sophisticated techniques are expected to: (i) have a higher accuracy in questions that these techniques are capable of answering, and (ii) be able to answer questions that the ones presented here are not, such as the ones requiring conditioned thinking

Chapter 8

Related Work

In this chapter we discuss some related topics. We start with Question Answering (QA) in Section 8.1, describing the characteristics of QA systems and a Portuguese competition of QA. Next, in Section 8.2, we discuss chatbots. Following, some QA tests that were proposed as replacements for the Turing Test (Section 8.3). Finally, in Section 8.4 we present the Portuguese initiative Linguateca and how Portuguese became part of CLEF.

8.1 Question Answering

Question Answering (QA), viewed as a Information Retrieval related task, is defined by retrieving a small piece of text that answers a question [Voo99, MHP+00]. In order to do so usually systems have 3 major phases: question classification, answer generation and answer classification.

The first of these phases, question classification, is responsible for identifying what a question asks for, that is, if the question begins with "who", then the answer must be a person or institution, "where" asks for a location; this was the strategy for question classification used generally by the systems attempting a competition of open-ended QA [Voo99] (TREC-8), but it did not suffice for all types of questions. In [MHP+00] — hereafter Lasso — the authors divided this phase in 3 minor stages: (i) question type — *who, what, where, why, how*; (ii) answer type — for some are ambiguous, viz. what, how; finally, (iii) question focus — the word or words that disambiguate the question.

The answer generation phase in TREC-8 was done mostly using Information Retrieval to search for relevant documents and then shallow parsing them to identify entities of the same type of the demanded by the question. Lasso also retrieves documents based on keywords, but ensuring that all keywords are present in the retrieved documents, then they process a operation called PARAGRAPH with an argument n, this function filters the document for relevant snippets of text, it searches for the keywords in a way that they are separated in the text by less than n paragraphs.

Finally, systems at TREC-8 generally compared if an entity identified is of the same type as the question requires and if it is close to words of the question, if it is, then the system returns that entity as response. Lasso uses a semantic parser in the selected paragraphs to identify a candidate-answer, this parser has the following characteristics: has grammar rules with heuristic rules to identify people's name, corporations, localization, dates, etc., and also has a lexical dictionary with semantic annotations derived from the WordNet [Mil95]. The next step is to establish an answer window for each candidate-answer and compute seven different metrics for each window, these metrics are then combined and ranked from the highest to lowest.

The winner of TREC-8 short answers — strings limited to 50 bytes — was [SL00]. This algorithm had Mean Reciprocal Rank (MRR) of 0.660 using the top five candidate-answers. The long answer — 250 bytes limit — winner was Lasso [MHP+99] with 0.646 MRR, being 77.7% of the right answers inside the top five candidate-answers.

It is noteworthy that IBM competed at TREC-8 [PBCR00] and achieved, respectively, 0.319 and 0.430 MRR for short and long answers. Years later, IBM displayed at Jeopardy! a super-human performing QA algorithm called Watson [FBCC+10]. The structure behind Watson is the same as

the one presented in this section, but Watson relies on a massive parallel processing, using a myriad of algorithms for every question and then using an algorithm to combine all the answers along with their confidences.

The systems developed for the Japanese QA competition [SSK+14] described in Section 3.2 had an extra phase. This phase was responsible for determining if an answer generated by the algorithm has any relation with the question's candidate-answers.

Other interesting competition was Págico, which required answering questions written in Portuguese with a Wikipedia page containing the answer to that question [San12]. For instance:

Q: Which Brazilian grammarians addressed the issue of the "Brazilian language" ? [1]

A: https://pt.wikipedia.org/wiki/Antenor_Nascentes

Two systems participated, both used Information Retrieval. The first [Car12], does the following: (1) transforms the question into a SPARQL query using handcrafted heuristics; (2) queries it to a multilingual ontology of structured knowledge, extracted from the Wikipedia, called dbpedia database [ABK+07]; (3) finally, the system retrieves information based on the answer of this database. The second system [ROG12] augmented the questions with synonyms of the identified noun and verbal phrases using WordNet. It also used part-of-speech tagging to identify what is being asked. The former achieved 8% accuracy, the latter, 12%.

8.2 Chatbots

Chatbots are programs that talk to humans. Ideally, one desirable chatbot characteristic would be the capacity to understand text — a message received — and respond accordingly. In other words, understand what is being stated and generate a proper answer. One of the first chatbots of history was created between 1964-1966 by Weizenbaum [Wei76]. The author says he had created a computer program that could "talk" — by typewriting — in English, he called this computer program Eliza.

Eliza used a two-tier architecture. The first one was responsible for language analysis. The second tier had a set of rules that enabled the program to give answers based on keywords used by the human interlocutor and to "improvise" about a given theme. This set of rules was called a script — for it was analogous to those of acting —, each script allowed Eliza to have a specific role on a conversation about a specific theme.

One of the scripts was called Doctor, because it tried to simulate a psychiatrist. Its main strategy was to make nominal changes on the phrase said by the interlocutor and return this altered phrase to him. Usually this new phrase would be a question — an example of conversation with this script can be seen in [Wei76, p. 3] or [Sag75, p. 10]. Weizenbaum stated that people became deeply connected with the program when talking to it and that they believed faithfully that it was a person, telling even their most intimate thoughts.

In 1991, the winner at the first Loebner Prize — (an attempt to run) a simplified Turing Test where algorithms had to converse about predetermined themes — was the program that competed on the topic: whimsical conversation. When the computer answered something unintelligible people tended to think that it was because of the theme or that they did not understand what was being stated. After the event it was said that little progress was made in those twenty-five years after Eliza [Shi94].

Recently, in Turing Test 2014 [Uni15, WS16], a chatbot called Eugene Goostman drew attention because it fooled 33% of the event's judges, people unfamiliar with Artificial Intelligence. As Doctor played the role of a psychiatrist, Eugene simulated a foreign teenager not fluent in English, thus misleading the judges to overlook its incapacity to answer adequately.

[1] Our translation, the original question is: Que gramáticos brasileiros se pronunciaram sobre a questão da "língua brasileira"?

8.3 Replacements for the Turing Test

Even though the Turing Test was very important, it was put aside by the community and now there are many proposals of replacements for the Turing Test in the literature. Some of these replacements are based on Question Answering and in Multiple-Choice Question Answering.

Levesque proposed a test where the algorithm has to think over a sentence and answer a binary question, this test was named Winograd Scheme Challenge [Lev11]. The sentence, usually a small one written in natural language, English, should contain an ambiguity that can only be resolved using commonsense reasoning. Examples of question are:

1. Question: The trophy would not fit in the brown suitcase because it was too big. What was too big?

   Answer 1: The Trophy

   Answer 2: The Suitcase

2. Question: Paul tried to call George on the phone, but he was not available. Who was not available?

   Answer 1: Paul

   Answer 2: George

The author proposes that for every question there should be an alternative version in which all occurrences of one world are changed and thus modifying the correct answer. For example:

1. Normal Question: Paul tried to call George on the phone, but he was not *available*. Who was not *available*?

   Answer 1: Paul

   Answer 2: George

2. Alternative Question: Paul tried to call George on the phone, but he was not *successful*. Who was not *successful*?

   Answer 1: Paul

   Answer 2: George

With this approach the author argues that having access to a large corpus would not help a system to "cheat" and find the answer based in statistical information rather than resolving the ambiguity using commonsense reasoning. In this case the author says that the needed is "background knowledge".

By its nature, the test requires the creation of a significant base of questions — and this responsibility was entrusted to the community. The Winograd Scheme Challenge does not require a specialist to interpret the performance of a program, because the performance is based on the number of right answers in a test. A test is generated by randomly choosing questions — normal or alternative versions — on the database. The problem with this evaluation system is that an answer is the right one if *normally* a human would choose it, thus requiring experimentation and repetition to bypass the lack of objectivity in the definition of solutions.

In [Dav16] Davis argues that standardized tests — such as the SAT, New York Regents 4th Grade Science Test, the Center Test and the ENEM — are created to be difficult for people, but not necessarily for computers, and what is difficult for one is not necessarily difficult for the other.

Davis states that standardized tests frequently ask for things that are easy for computers in the actual state of the art: the definition of a jargon (terminology), create groups of individuals

and instance relations between groups (taxonomy); retrieve standard formulae and use them in calculations (exact calculation).

He thus proposes creating the SQUABU (Science Questions Appraising Basic Understanding), a curated dataset of multiple-choice questions using both High School and Primary School Science tests. The primary school questions should require understanding of time, causality, human body, sets of objects, simple cases of inductive arguments of unknown length, combining facts and also recognizing impossible scenarios; the high school questions, on the other hand, should be more related to scientific methodology, including interpreting real world phenomena and laboratory experiments.

The author states that although the issue of time — representing temporal information and temporal reasoning — being almost a solved problem [Rei01], it tends to be neglected in practice, like in NELL [MCH+15] and ConceptNet [HSA07]. Similarly, he states that inductive arguments of indeterminate length are presented in the literature of software verification, but it is not integrated with AI. Examples of questions that require each knowledge are:

> Time: Sally's favorite cow died yesterday. The cow will probably be alive again (A) tomorrow; (B) within a week; (C) within a year; (D) within a few years; (E) The cow will never be alive again.
>
> Undetermined Length Induction: Mary owns a canary named Paul. Does Paul have any ancestors who were alive in the year 1750? (A) Definitely yes. (B) Definitely no. (C) There is no way to know

Causality and Pointless Scenarios are to be tested using so obvious scenarios that no one would have stated them in the internet, hence no corpus would help the algorithm to retrieve the right answer. Examples of questions:

> Pointless Scenario: Is it possible to fold a watermelon?
>
> Causality: Suppose you have two books that are identical except that one has a white cover and one has a black cover. If you tear a page out of the white book what will happen? (A) The same page will fall out of the black book. (B) Another page will grow in the black book. (C) The page will grow back in the white book. (D) All the other pages will fall out of the white book. (E) None of the above.

Questions about the human body give humans an advantage; the author states that even basic spatial properties may be hard to AI, the same goes for sets of (undetermined number of) objects; putting facts together may be difficult when the information come from different sources. Examples of questions are:

> Putting Facts: George accidentally poured a little bleach into his milk. Is it OK for him to drink the milk, if he's careful not to swallow any of the bleach?
>
> Human Body: Can you see your hand if you hold it behind your head?
>
> Spacial Properties: When Ed was born, his father was in Boston and his mother was in Los Angeles. Where was Ed born? (A) In Boston. (B) In Los Angeles. (C) Either in Boston or in Los Angeles. (D) Somewhere between Boston and Los Angeles.
>
> Sets of Objects: There is a jar right-side up on a table, with a lid tightly fastened. There are a few peanuts on the table. Joe picks up the jar and shakes it up and down, then puts it back on the table. At the end, where, probably, are the peanuts? (A) In the jar. (B) On the table, outside the jar. (C) In the middle of the air.

The focus of the High School version is to elevate the understanding of the relation between formal science and commonsense reasoning. The domain of this test plays an important role, the author states that it must be a domain where the two knowledge relate to each other deeply and evidently. Davis proposes three domains: chemistry experiments, astronomy and problems in everyday settings. Examples of questions are:

> Chemistry: In the Millikan oil-drop experiment, a tiny oil drop charged with a single electron was suspended between two charged plates. The charge on the plates was adjusted until the electric force on the drop exactly balanced its weight. How were the plates charged? (A) Both plates had a positive charge. (B) Both plates had a negative charge. (C) The top plate had a positive charge, and the bottom plate had a negative charge. (D) The top plate had a negative charge, and the bottom plate had a positive charge. (E) The experiment would work the same, no matter how the plates were charged.
>
> Astronomy: Does it ever happen that someone on Earth sees an eclipse of the moon at noon?
>
> Everyday: Suppose that you have a large closed barrel. Empty, the barrel weighs 1 kg. You put into the barrel 10 gm of water and 1 gm of salt, and you dissolve the salt in the water. Then you seal the barrel tightly. Over time, the water evaporates into the air in the barrel, leaving the salt at the bottom. If you put the barrel on a scales after everything has evaporated, the weight will be (A) 1000 gm (B) 1001 gm (C) 1010 gm (D) 1011 gm (E) Water cannot evaporate inside a closed barrel.

Comparing SQUABU with Winograd Scheme Challenge we can see that the former does not systematize a number of alternatives like the latter, also while the latter has small phrases the former does not have a usual length. These differences aside, both focus on developing an uncheatable test that requires further development of AI, the Winograd Scheme Challenge aims at commonsense reasoning, SQUABU, on the other hand, has some granulated categories. Note that SQUABU requires also the creation of questions, probably by the community.

In [PM16] the authors describe the test they are using at Center for Brains, Minds, and Machines at MIT. They call it the Turing++ test, the name is due to additional requirements that they have that are not present in the original test. In principle the format of the test is equal to the Turing Test, that is, open-ended questions, the difference is that the Turing++ questions are on images and the questions are "who is there?", "what is the person doing?", "what does the boy think about the girl?", etc. The additional requirement is that not only the answer should be human-like, but also the process to generate the answer must be analogous to the human process.

Different from the previous two, the Turing++ requires also advancements of another areas, such as physiology and neurology, so that it becomes possible to compare the process of the algorithm to those of a human. This test does not have the characteristic, like the others, of trying to develop a lacking point of AI, instead it urges the field to diminish the difference between humans and AI.

More tests are presented in [JY16, Jun16, ABC16], they are based on Gardner's theory of multiple intelligence [Gar83], each of these tests aims at extending the Turing Test to other(s) type(s) of intelligence(s).

8.4 Linguateca and CLEF

The Conference and Labs of the Evaluation Forum (CLEF) [PBC+04] is an European Cross-language Information Retrieval project that promotes the development of different strategies and the usage of different languages. Since its creation the project has organized a series of competitions with specialized tracks. In these tracks systems compete in different tasks, such as Monolingual Information Retrieval, Multilingual Information Retrieval and Question Answering.

The Linguateca[2] initiative was created in order to foster research on Portuguese language processing. The initiative's goals are to inform, create and disseminate resources made in Portuguese, in addition to promoting evaluation contests in Portuguese [SR05]. Some of the resources made available are: a large corpus of newspaper articles, the CETEMPúblico [SR01], and the Floresta Sinta(c)tica treebank [ABHS02], etc. Moreover, the website is not limited to making resources available, it also contains information about people, groups, projects, publications and companies that work with Portuguese language processing.

[2] https://www.linguateca.pt/

Linguateca joined the CLEF in 2004 making Portuguese one of the languages available at the competitions. In [SR05] and [RS07] the authors describe the results achieved from 2004 to 2006, the importance of joining CLEF and some issues that needed to be overcome for the inclusion of Portuguese resources in the competitions. They discuss about the preparations to include Portuguese in the IR and QA tracks. One point discussed is the topic division of the competition, for instance: it is not trivial to determine if an event is an international or a national event, as it does not depend on the event itself, but on its media coverage. To deal with this problem the authors propose another classification for topics: cyclic events, once-only events, states of broader events, impact measures and atemporal subjects. For the QA track the authors state the challenges of devising a QA dataset, such as creating natural questions that are not too difficult and that have a straightforward answer, finding all possible answers in the data collection and — because of CLEF — translating the question and answers properly to and from English. As with IR, question classification was controversial, since the classification had nothing to do with linguistic properties of the question, but rather with semantic properties of the answer, such as if the answer is a person, a location, object, etc. To avoid this, the authors proposed classifying questions according to the linguistic entity expected in the right answer. Additionally, the authors argue against questions on definitions for this task, as they overlap with other types of question and are not suitable for the database — which consists of news articles from newspapers. Finally, the authors highlight the importance of selecting (or creating) proper questions to maximize the utility of the dataset, bearing in mind the collection that is being used and integrating more Question Answering and Information Retrieval. The dataset and database used were named CHAVE and are available at https://www.linguateca.pt/CHAVE/.

Chapter 9

Conclusion

In this chapter we review our goals, contributions and the results achieved in this work (Section 9.1). To conclude, we present some future work (Section 9.2).

9.1 Final Considerations

In this thesis we presented the creation of a dataset of questions taken from the Ensino Nacional do Ensino Médio. This dataset is composed of a wide variety of subjects — ranging from physics to questions of poem interpretation — arranged in four different major areas: Mathematics, Science, Languages and Human Sciences. In this creation we manually tagged each question based on the type of knowledge it requires: Text Comprehension, Encyclopedic Knowledge, Image Comprehension, Domain Specific knowledge and Mathematical Reasoning. Additionally we tagged if the question has an image in its scope and if it requires understanding about a chemical element. This dataset is made available publicly[1].

Following we presented the performance of algorithms based on Information Retrieval[2] and Word Embedding[3]. Information Retrieval answers questions based on how similar a query is compared to the documents of a knowledge database; Word Embedding, on the other hand, uses the cosine similarity between the representations of question and candidate-answer.

To enhance the performance of these two approaches we combined them with augmentations using WordNet's relations: Synonym, Hypernym, Hyponym and Holonym, alternatively we also used close words found by GloVe; we discovered that Hypernym provide additional information that most often contribute to the performance of the algorithms, meaning that knowing hierarchical relation between concepts is of great relevance. We would like to stress that the WordNet[4] used is not a Brazilian ontology, it translates a word to English, gets the relation of the words in English and translates it back to Portuguese. Additionally, augmenting using GloVe was worse than not augmenting for most cases.

We showed that Information Retrieval with small databases profits from augmentation: the smaller database (Header) went from 20.9% to 24.13% accuracy, the medium database (ENEM) went from 23.37% to 25%; but the bigger database (Wikipedia[5], 1.5G of plain text) achieved 26.9% and augmentation hurt its performance; alternatively, combining augmentation with a sequential usage of the Header and ENEM databases achieved 25.28%.

We also investigated the usage of Word Embedding with the strategy of summing the vectors of the words in order to create a vector representation of a larger piece of text. We found that it is usually better to make the cosine similarity of the alternatives with the statement, ignoring the text given in the header. It was shown that when solving the ENEM the CBOW model of

[1] Available at: https://www.ime.usp.br/~ddm/enem/
[2] We used the software Lucene version 6.4. The software is available at: https://lucene.apache.org/core/
[3] The trained vectors were obtained at: http://nilc.icmc.usp.br/embeddings
[4] We used the Open Multilingual WordNet available at python NLTK.
[5] Dump of January 2018.

Word2Vec and Wang2vec are better than their Skip model, but regarding FastText the opposite is true. Moreover, in our task FastText (28% ±4.5) was better than the other types of embedding. Another contribution of this work is that we came upon the fact that higher dimensions of vectors (more than 300) tended to do better in solving the exams, the same happened in Semantic Similarity evaluation of [HFS+17]. For them FastText was the best in syntactic analogies and Skip was better than CBOW, for us FastText is also the best, but CBOW tends to do better than Skip — except in the case of FastText.

It was shown that using a greedy search it is possible to find an ensemble of algorithms that outperformances the individual performance of its components, thus achieving 29% accuracy with 2.2 of standard deviation. The accuracy of the Ensemble approach per knowledge tag is:

Pure Text Comprehension: 33.5%;

Text Comprehension: 32.5%;

Pure Encyclopedic Knowledge: 37.7%;

Encyclopedic Knowledge: 32.3%;

Pure Domain Specific: 16.6%;

Domain Specific: 28.9%.

We advocate that the increase in Text Comprehension over an early work [SM17] is due to combination of different methods of Word Embedding, each capturing a nuance of the text. Domain Specific knowledge is present in our methods only in correlation of the words, this is one reason why the methods may have poor performance in questions of pure Domain Specific. Additionally, while we may be far from Aristo's 71.3% accuracy solving elementary science questions [CEK+16] — where its Information Retrieval Solver achieves almost 60% —, the results presented by Toudai [FKKM14] in the English subject (26%) is close to ours, but they achieve 41.4% when evaluating together English, Japanese, Japanese History and World History.

Regarding the disparity between Aristo's performance and ours, we tested another common algorithm, namely, extracting features through Word Embedding and using them as input to a Support Vector Machine. While Aristo achieves 55.4% using four groups of vectors trained in different domain specific corpora, we achieved 25% accuracy using 35 different vectors trained in a heterogeneous corpora. This shows that either the method is not completely generalizable or it is highly dependant on domain specificity.

Finally, analyzing the questions that no algorithm got right we came upon six different groups of questions, five that require capacities that the algorithms do not have and one that is simply the questions that they are supposed to answer but got wrong nevertheless. The six groups are: (1) knowing key concepts; (2) identifying emotions; (3) understanding text genre; (4) conditioned thinking; (5) simple flaws; (6) others. These groups are in concordance with the capacities that other proposed Artificial Intelligence tests desire to improve. Consequently we expect that by using the ENEM dataset Portuguese researchers can contribute to problems working on their first language.

Thus we conclude that the ENEM is a complex and appealing task to be solved: appealing because it is a real word task that is experienced by almost every student and highly valued in hodiernal society; complex because in order to achieve human-level performance it requires the usage of different types of knowledge and certain desirable characteristics of AI systems, such as identifying consequences, relating concepts, recognizing the main point of a text and what emotions a piece of text conveys.

9.2 Future Work

The work considered here could be extended to other university entrance exams, not only from Brazil but from other countries, which would indicate the generalization ability of the proposed

methods. Different Knowledge Bases could be used with Information Retrieval. In special, it would be interesting to use school-oriented texts, such as textbooks or articles crawled from websites that have test-driven content. Other approaches using Word Embedding could be tested, such as: specializing the embedding in solving an exam and then using it with the approaches described in this work. Alternatively, a Neural Network could be trained to solve the exam having the embedding as middle layer. More sophisticated solvers, that use structured knowledge, could be implemented and tested — like the TableILP from Aristo. Also, it would be relevant to present the performance, in our dataset, of programs that try to identify entailment, emotions and statistical connections. Finally, some different strategies could be used to combine the algorithms presented here. These strategies could, for instance, use features of the question, like the knowledge tag, or even learn the features that will be used.

Bibliography

[ABC16] Sam S. Adams, Guruduth Banavar and Murray Campbell. I-athlon: Towards a multi-dimensional Turing test. *AI Magazine*, 37:78–84, 2016. Appears in pg. 65

[ABHS02] Susana Afonso, Eckhard Bick, Renato Haber and Diana Santos. Floresta sintá(c)tica: A treebank for portuguese. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages: 1698–1703, 2002. Appears in pg. 65

[ABK+07] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International Semantic Web Conference*, pages: 722–735, 2007. Appears in pg. 62

[BDK14] Marco Baroni, Georgiana Dinu and Germán Kruszewski. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages: 238–247, 2014. Appears in pg. 7

[BGJM17] Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. Appears in pg. 8, 9

[Bra16] Portal Brasil. Mais de 9,2 milhões se inscreveram no ENEM 2016. (http://www.brasil.gov.br/educacao/2016/05/mais-de-9-2-milhoes-de-candidatos-se-inscreveram-no-enem), 2016. [Online; accessed April-29-2018]. Appears in pg. 1

[Car12] Nuno Cardoso. Medindo o precipício semântico. *Linguamática*, 4:41–48, 2012. Appears in pg. 62

[CCE+18] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *Unpublished*, 2018. Appears in pg. 12

[CEK+16] Peter Clark, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Turney and Daniel Khashabi. Combining retrieval, statistics, and inference to answer elementary science questions. In *Proceedings of the 30th Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*, pages: 2580–2586, 2016. Appears in pg. 2, 3, 11, 26, 30, 68

[CZW+16] Gong Cheng, Weixi Zhu, Ziwei Wang, Jianghui Chen and Yuzhong Qu. Taking up the gaokao challenge: An information retrieval approach. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages: 2479–2485, 2016. Appears in pg. 1, 15

[Dav16] Ernest Davis. How to write science questions that are easy for people and hard for computers. *AI Magazine*, 37:13–22, 2016. Appears in pg. 63

[FBCC+10] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A. Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John Prager, Nico Schlaefer and Chris Welty. Building Watson: An overview of the DeepQA Project. *AI Magazine*, 31:59–79, 2010. Appears in pg. 3, 30, 61

[FJHP+15] Daniel Fried, Peter Jansen, Gustave Hahn-Powell, Mihai Surdeanu and Peter Clark. Higher-order lexical semantic models for non-factoid answer reranking. *Transactions of the Association for Computational Linguistics*, 3:197–210, 2015. Appears in pg. 27

[FKKM14] Akira Fujita, Akihiro Kameda, Ai Kawazoe and Yusuke Miyao. Overview of Todai Robot Project and evaluation framework of its NLP-based problem solving. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages: 2590–2597, 2014. Appears in pg. xiii, 1, 14, 30, 68

[Gar83] Howard Gardner. *Frames of Mind: the theory of multiple intelligences.* Basic Books, 1983. Appears in pg. 65

[HFS+17] Nathan Hartmann, Erick Rocha Fonseca, Christopher Shulby, Marcos Vinícius Treviso, Jéssica Silva and Sandra M. Aluísio. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. In *Proceedings of the 11th Brazilian Symposium in Information and Human Language Technology*, pages: 122–131, 2017. Appears in pg. 9, 27, 68

[HSA07] Catherine Havasi, Robert Speer and Jason B. Alonso. Conceptnet 3: A flexible, multilingual semantic network for common sense knowledge. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages: 261–267, 2007. Appears in pg. 64

[JSC14] Peter Jansen, Mihai Surdeanu and Peter Clark. Discourse complements lexical semantics for non-factoid answer reranking. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages: 977–986, 2014. Appears in pg. 12

[Jun16] Charles. L. O. Junior. Why we need a physically embodied Turing test and what it might look like. *AI Magazine*, 37:55–62, 2016. Appears in pg. 65

[JY16] William Jarold and Peter Z. Yeh. The social-emotional Turing challenge. *AI Magazine*, 37:31–38, 2016. Appears in pg. 65

[KBG+15] Tushar Khot, Nirajan Balasubramanianm, Eric Gribkoff, Ashish Sabharwal, Peter Clark and Oren Etzioni. Markov logic networks for natural language question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages: 685–694, 2015. Appears in pg. 11

[LDBT15] Wang Ling, Chris Dyer, Alan W. Black and Isabel Trancoso. Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages: 1299–1304, 2015. Appears in pg. 8, 9

[Lev11] Hector J. Levesque. The Winograd schema challenge. In *Proceedings of the 10th International Symposium on Logical Formalization on Commonsense Reasoning*, pages: 63–68, 2011. Appears in pg. 63

[McC16] Chris McCormick. Word2vec tutorial. http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/, 2016. Accessed: 2017-08-03. Appears in pg. xi, 8

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. Efficient estimation of word representations in vector space. *Unpublished*, 2013. Appears in pg. 7

[MCH+15] Tom M. Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Justin Betteridge, Andrew Carlson Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Ndapa Nakashole, Emmanouil Antonios Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard Wang, Derry Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves and Joel Welling. Never-ending learning. In *Proceedings of the 29th Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*, pages: 2302–2310, 2015. Appears in pg. 64

[MHP+99] Dan I. Moldovan, Sanda M. Harabagiu, Marius Pasca, Rada Mihalcea, Richard Goodrum, Roxana Girju and Vasile Rus. LASSO: A tool for surfing the answer net. In *Proceedings of the 8th Text Retrieval Conference*, pages: 175–184, 1999. Appears in pg. 61

[MHP+00] Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum and Vasile Rus. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages: 563–570, 2000. Appears in pg. 61

[Mil95] George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41, 1995. Appears in pg. 61

[MK13] Yusuke Miyao and Ai Kawazoe. University entrance examinations as a benchmark resource for NLP-based problem solving. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*, pages: 1357–1365, 2013. Appears in pg. xi, xi, 1, 12, 13, 20

[MMS+14] Suguru Matsuyoshi, Yusuke Miyao, Tomohide Shibata, Chuan-Jie Lin, Cheng-Wei Shih, Yotaro Watanabe and Teruko Mitamura. Overview of the NTCIR-11 recognizing inference in text and validation (RITE-VAL) task. In *Proceedings of the 11th National Institute of Informatics Testbeds and Community for Information Access Research Conference*, pages: 223–232, 2014. Appears in pg. 13

[MRS08] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. Appears in pg. xi, 5, 6, 7

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages: 3111–3119, 2013. Appears in pg. 2, 7

[OZLL15] Xi Ouyang, Pan Zhou, Cheng Hua Li and Lijun Liu. Sentiment analysis using convolutional neural network. In *Proceedings of the 15th International Conference on Computer and Information Technology*, pages: 2359–2364, 2015. Appears in pg. 59

[PBC+04] Carol Peters, Martin Braschler, Khalid Choukri, Julio Gonzalo and Michael Kluck. The future of evaluation for cross-language information retrieval systems. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages: 841–844, 2004. Appears in pg. 65

[PBCR00] John Prager, Eric Brown, Anni Coden and Dragomir Radev. Question-answering by predictive annotation. In *Proceedings of the 23rd Annual International ACM Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval*, pages: 184–191, 2000. Appears in pg. 61

[PM16] Tomaso A. Poggio and Ethan Meyers. Turing++ questions: A test for the science of (human) intelligence. *AI Magazine*, 37:73–77, 2016. Appears in pg. 65

[PSM14] Jeffrey Pennington, Richard Socher and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages: 1532–1543, 2014. Appears in pg. 8

[RD06] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006. Appears in pg. 11

[Rei01] Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001. Appears in pg. 64

[RN02] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2002. Appears in pg. 2, 5

[ROG12] Ricardo Rodrigues, Hugo G. Oliveira and Paulo Gomes. Uma abordagem ao Págico baseada no processamento e análise de sintagmas dos tópicos. *Linguamática*, 4:41–48, 2012. Appears in pg. 62

[RS07] Paulo Rocha and Diana Santos. CLEF: Abrindo a porta à participação internacional em avaliação de RI do portugês. In Diana Santos, editor, *Avaliação Conjunta: Um Novo Paradigma no Processamento Computacional da Língua Portuguesa*, pages: 143–158. IST Press, 2007. Appears in pg. 66

[Sag75] Carl Sagan. In praise of robots. *Natural History*, 84:8–23, 1975. Appears in pg. 62

[San12] Diana Santos. Porquê o Págico? Razões para uma avaliação conjunta. *Linguamática*, 4:1–8, 2012. Appears in pg. 62

[Shi94] Stuart M. Shieber. Lessons from a restricted Turing test. *Communications of the ACM*, 37:70–78, 1994. Appears in pg. 62

[SL00] Rohini Srihari and Wei Li. A question answering system supported by information extraction. In *Proceedings of the 6th Conference on Applied Natural Language Processing*, pages: 166–172, 2000. Appears in pg. 61

[SM17] Igor Cataneo Silveira and Denis Deratani Mauá. University entrance exam as a guiding test for artificial intelligence. In *Proceedings of the 6th Brazilian Conference on Intelligent Systems*, pages: 426–431, 2017. Appears in pg. 68

[SR01] Diana Santos and Paulo Rocha. Evaluating CETEMPublico, a free resource for portuguese. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages: 442–449, 2001. Appears in pg. 65

[SR05] Diana Santos and Paulo Rocha. The key to the first CLEF with portuguese: Topics, questions and answers in CHAVE. In *Proceedings of the 5th Workshop of the Cross-Language Evaluation Forum*, pages: 821–832, 2005. Appears in pg. 65, 66

[SSK+14] Hideyuki Shibuki, Kotaro Sakamoto, Yoshinobu Kano, Teruko Mitamura, Madoka Ishioroshi, Kelly Y. Itakura, Di Wang, Tatsunori Mori and Noriko Kando. Overview of the NTCIR-11 QA-lab task. In *Proceedings of the 11th National Institute of Informatics Testbeds and Community for Information Access Research Conference on Evaluation of Information*, pages: 518–529, 2014. Appears in pg. 14, 62

[SWY75] G. Salton, A. Wong and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18:613–620, 1975. Appears in pg. 5

[Uni15] Turing test success marks milestone in computing history. (http://www.reading.ac.uk/news-and-events/releases/PR583836.aspx), 2015. [Online; accessed April-23-2017]. Appears in pg. 62

[Voo99] Ellen M. Voorhees. The TREC-8 question answering track report. In *Proceedings of 8th Text REtrieval Conference*, pages: 77–82, 1999. Appears in pg. 61

[Wei76] Joseph Weizenbaum. *Computer Power and Human Reason: From Judgment to Calculation*. W. H. Freeman & Co., 1976. Appears in pg. 62

[WS16] Kevin Warwick and Huma Shah. *Turing2014: Tests at The Royal Society*, pages: 171–186. Cambridge University Press, 2016. Appears in pg. 62

[Yua16] Tao Yuan. Number of gaokao students dwindling overall. (http://english.cctv.com/2016/06/07/VIDElr9mDklWGIfFywe1zPLK160607.shtml), 2016. [Online; accessed April-29-2018]. Appears in pg. 1