Inconsistency Detection in Job Postings

Joana Urbano[1]
skeeled, Bascharage, Luxembourg
Artificial Intelligence and Computer Science Laboratory (LIACC), Porto, Portugal

Miguel Couto
skeeled, Bascharage, Luxembourg

Gil Rocha
Faculty of Engineering, University of Porto, Portugal
Artificial Intelligence and Computer Science Laboratory (LIACC), Porto, Portugal

Henrique Lopes Cardoso
Faculty of Engineering, University of Porto, Portugal
Artificial Intelligence and Computer Science Laboratory (LIACC), Porto, Portugal

Abstract

The use of AI in recruitment is growing and there is AI software that reads jobs' descriptions in order to select the best candidates for these jobs. However, it is not uncommon for these descriptions to contain inconsistencies such as contradictions and ambiguities, which confuses job candidates and fools the AI algorithm. In this paper, we present a model based on natural language processing (NLP), machine learning (ML), and rules to detect these inconsistencies in the description of language requirements and to alert the recruiter to them, before the job posting is published. We show that the use of an hybrid model based on ML techniques and a set of domain-specific rules to extract the language details from sentences achieves high performance in the detection of inconsistencies.

2012 ACM Subject Classification Computing methodologies → Natural language processing; Applied computing → Enterprise ontologies, taxonomies and vocabularies

Keywords and phrases NLP, Ambiguities, Contradictions, Recruitment software

Digital Object Identifier 10.4230/OASIcs.LDK.2021.25

Funding *Gil Rocha*: Gil Rocha is supported by a PhD grant (with reference SFRH/BD/140125/2018) from Fundação para a Ciência e a Tecnologia (FCT).
Henrique Lopes Cardoso: This research is partially supported by LIACC (FCT/UID/CEC/0027/2020), funded by FCT.

Acknowledgements We want to thank Catarina Correia for the valuable contribution she made in the initial phase of this project.

1 Introduction

AI-based recruitment tools automate parts of the recruitment process. One of these parts is the prescreening of candidates, where a given applicant is matched against a job description using a machine learning algorithm that predicts whether or not this applicant is suited for further analysis by the recruiter.

Recruitment software often allows job information to be entered by the recruiter in textual descriptions of the job requirements and/or in structured fields that the recruiter must fill-in but that may or may not be visible to the applicant [1]. As an example, the recruiter may ask in the textual description for "good knowledge of English" and then fill-in structured fields on language requirements with "English" and language level "B2". As another example, the

[1] Corresponding author

recruiter may write "very experienced in the astrophysics domain" in the textual description and then select "10+ years of experience" in a structured field. This approach suits well recruiters, as they are very used to traditional textual job descriptions; however, they need to be able to fill-in structured fields in a way that is consistent with the textual description. The problem that arises is that there are often ambiguities and contradictions between the textual descriptions and the structured fields, which makes it challenging for AI-based algorithms, applicants and fellow recruiters to correctly interpret the job offer.

Based on the knowledge gathered by analysing the most frequent inconsistencies in a corpus of job descriptions, we developed a Natural Language Processing (NLP) model that uses both Machine Learning (ML) and rules to detect contradictions and ambiguities in job descriptions. With such a model, recruitment software is able to alert the recruiter of these inconsistencies before a job posting is published. In this paper, we present this model, focusing on the description of language requirements in job postings written in the English language. The proposed approach is currently being extended into the detection of inconsistencies in other components of job descriptions, such as the description of fields of study. The main contributions of this paper are: 1) the identification of the most common types of ambiguities and contradictions that occur when describing language requirements in job descriptions, resulting from our thorough analysis of a corpus of about 1,500 job descriptions comprising different roles, industries and clients; and 2) our NLP-based model that uses ML and a small set of rules, that has shown high performance in the detection of inconsistencies in job postings.

2 Contradictions and Ambiguities

When addressing the inconsistencies that may arise between the textual descriptions and the structured fields of a job posting, we distinguish between *contradictions* and *ambiguities*. In this section, we provide a common definition for these two types of inconsistencies and then we review how the NLP community addresses the detection of such inconsistencies.

Contradictions

We consider that a *contradiction* happens between two pieces of information when the probability of both being simultaneously true is extremely unlikely [12]. As an example applied to the description of language requirements in job descriptions, there is a contradiction when the recruiter writes that "it is required proficiency in English" and then s/he sets the English language level to A2, because these two pieces of information are not pragmatically aligned, as A2 is associated to a basic understanding of a language [13].

There are different types of contradictions being addressed by the NLP community. Marneffe, Rafferty, and Manning propose a typology of contradiction classes including *antonym*, *negation*, *numeric*, *factive*, *structural*, *lexical* and *world-knowledge* types [12].

Ambiguities

We consider that one sentence is *ambiguous* when it can have more than one possible interpretation [20], which can cause uncertainty to the reader. In this sense, the ambiguity is self-contained in a textual sentence and does not depend on the relation of this sentence with other fields, contrary to what usually happens with contradictions. An example of such an ambiguity is the sentence "You must be English/French fluent". Here, it is not clear if the candidate must be fluent in both English and French, or if it is enough to be fluent in just one of these two languages.

There are different types of ambiguities that are addressed by the NLP community, although most of them arise when a word or sequence of words have different meanings, in the same or in different contexts, with no other differences at the grammatical level (e.g., *lexical*, *pragmatic*, *semantic*), when they allow for more than one grammatical structure or different groupings of adjacent words (e.g., *syntactic*, *surface structuring*), or where the sequence of words admit borderline cases (*vagueness*) [4, 17, 20].

2.1 Related Work

The literature for detecting contradictions in text using NLP is still relatively scarce. Marneffe, Rafferty and Manning [12] present a typology of contradictions and propose an NLP-based system to automatically detect contradictions between two different sentences. Their approach converts both sentences into typed dependency graphs that are aligned in order to extract different contradiction-related features. Then, a logistic regression model is trained over these features to learn the contradiction value. Li, Qin and Liu [10] propose a convolutional neural network model to learn contradiction-specific word embedding (CWE), arguing that the use of CWE outperforms context-based word embeddings in the detection of contradictions. A different approach is provided by Pham, Nguyen and Shimazu [14], who propose a rule-based system to detect contradictions based on shallow semantic representations and binary relations extracted from sentences. Finally, Dragos [6] proposes a system to detect contradictions between two sentences that uses fuzzy semantics and that implies the estimation of the certainty of the statements, allowing to distinguish between contradictions derived from disagreement and those derived from conflictual opinions.

The use of NLP to detect ambiguities in text has some expressiveness in the Requirements Engineer domain. Gleich, Creighton, and Kof [9] use Part-Of-Speech (POS) tagging to detect the use of passive voice, adjectives and adverbs and then check for ambiguity patterns at the word level in requirements documents. Rosadini et al. [17] use POS tagging and shallow parsing to design patterns used to detect anaphoric ambiguities, vague terms, passive voice, and undefined terms in manufacturing requirements documents. They conclude that ambiguities requiring domain knowledge are hard to detect using rule-based approaches. Sabriye and Zainon [18] also use POS to detect syntactic ambiguities in software requirements documents. More concretely, they consider that a given sentence is ambiguous if it generates more than one parse tree or if it contains any "AND" or "OR" conjunctions. Rojas and Sliesarieva [16] use syntactic parsing in conjunction with regular expressions to identify vague adverbs and other ambiguous phrase structures, as well as dictionaries (WordNet [7] and VerbNet [19]) to identify ambiguous verbs. Ferrari, Donasi and Gnesi [8] study how specific words from the Computer Science lexicon vary in terms of ambiguity in different domains. For this, they built specific word embeddings from distinct vector spaces constructed in different document categories (e.g, Electronic Engineering and Medicine) and measure the variation of meaning of the CS terminology within these categories.

Most of the approaches we overview in this section use NLP to build patterns to detect contradictions and ambiguities in a way that is independent of the domain. To the best of our knowledge, our approach is the first to address the detection of ambiguities and contradictions in language requirements of job descriptions and to propose a typology of contradictions and ambiguities in language requirements.

3 Inconsistency Model Architecture

In this section, we present the general architecture of our model to detect inconsistencies in the description of language requirements in job postings written in English.

3.1 Language Requirements Specification

In this work, we consider that a job description may specify zero or more *required* languages and zero or more *optional* languages. Optional languages are only described in the textual description of the job posting, whereas required languages may be specified both in the textual description and in the structured fields of the job posting. Moreover, we consider the possibility of specifying *alternative* languages. To better understand these concepts, let us analyze the following example of a textual description.

▶ **Example 1.** The candidate must have experience in the domain and a masters in biology, biochemistry or related areas. Be very organized. We expect good knowledge of English and similar knowledge of either French or Portuguese; German is considered an asset.

With this description, the recruiter wants candidates to have good level of, at least, two languages, one being English and the other being either French or Portuguese. French and Portuguese are then alternative languages, and the minimum number of required languages is two. The recruiter also indicates that language skills in German are optional. This information must also be defined in the following structured fields, for each one of the languages mentioned in the textual description (cf. Table 1):

- *language*: the two-letter language code [2]. Sentences such as "Any other language will be considered an asset" do not require any entry in structured fields, although some recruiters may enter specific optional languages.
- *level*: the minimum language level required for the language.[2]
  In this example, we assume that good knowledge of a language corresponds to a B2 level, but our model will allow for some degree of flexibility in the definition of language levels as different recruiters may have different understandings of how to assign these levels. The specification of a language level is mandatory, even if there is no reference to such in the textual description, as it happens with the definition of German, in this example.
- *optional*: this defines whether the language is considered optional or not.
- *alternative*: this applies to non-optional languages only, and allows to distinguish between the languages that the applicant must definitely know ("no") and those that are considered alternatives ("yes").

Table 1 Example of the definition of structured fields.

language	level	optional	alternative
en	B2	no	no
fr	B2	no	yes
pt	B2	no	yes
de	B1	yes	–
Required languages:	2		

[2] Levels A1 and A2 are basic levels, B1 and B2 intermediary, and C1 and C2 proficient levels [13].

Looking again at this example, an applicant with B2 or upper levels in English and French would fit the required languages, as would an applicant with B2 or upper levels in English and Portuguese, or even these levels in English, French and Portuguese. However, an applicant with B2 or upper levels in English and Spanish or in English and German would lack one required language. In the same way, an applicant with B1 or lesser in any of the non-optional languages would be considered unfit in terms of the language requirements.

As a final note, it is frequent that recruiters put less information on the textual description than in structured fields. As an example, structured fields may specify English and French as required languages but no references to these languages are made in the textual description. As another example, the structured fields may require English, French, and German as required languages, but the textual description only mentions that "a very good understanding of German is an absolute need". These two examples show a discrepancy between textual and structured fields, but we do not consider them as being contradictory or ambiguous in this work. However, if a non-optional language is specified in the textual description and it is not specified in the structured fields, this would be a contradiction.

3.2 Language-related Inconsistencies

We analyzed a corpus of more than 1500 job postings written in English by different recruiters of different companies, concerning different roles in very distinct domains and industries (e.g., automotive, restaurants, hotels and leisure, health, air and ground transportation, biotechnology, banking, education, to name just a few). After isolating the language-related sentences, we analyzed them manually looking for ambiguities and/or contradictions within textual descriptions or between the textual descriptions and the structured fields. Backed on prior work, this allowed us to build our own terminology for the most representative types of ambiguities and contradictions in language requirement descriptions, which we present next.

- **Language-not-specified contradiction.** This occurs when a given language is mentioned in the textual job description but is not listed in the correspondent structured fields for required or optional languages.
- **Language-not-required contradiction.** This occurs when a language is identified in the required languages structured fields but is referred to as an optional language in the textual description (e.g., listing French as required and then writing "French would be an advantage" in the textual description).
- **Language-not-optional contradiction.** This is the reverse of the previous contradiction: a language is listed as optional in the structured fields while being referred to as required in the textual description (e.g., listing Italian as optional and writing "Italian is a must" in the textual description).
- **Lexical contradiction.** This occurs when the required language levels described in job textual requirements are not aligned with the language levels specified in structured fields. An example of such a contradiction is when a textual description asks for "fluency in English" and the structured fields specify B1 as the minimum level accepted for English.
- **Numerical contradiction.** This occurs when the number of minimum required languages described in the textual description does not correspond to the one specified in the structured fields. An example is when the job mentions "You are fluent in English as well as either Norwegian or Swedish" and then specifies three minimum required languages in structured fields. In fact, the textual description only refers to two minimum required languages, one mandatory and the other selected from two alternative languages.

- **Alternative-language contradiction.** This occurs when the number of alternate languages is not the same in textual and structured fields. An example is when the textual description is "You are fluent in English as well as either Norwegian or Swedish" and then in the structured fields all languages are considered as non-alternative.
- **Ambiguity.** This occurs in sentences that can have more than one possible interpretation or that are somewhat vague. For example, sentence "Good knowledge of Portuguese and preferably good knowledge of Spanish" is not clear about whether the term "preferably" applies to the language (and then Spanish would be optional) or to the desired language level (i.e., Spanish would be required and the desired level would be good). As another example, the sentence "You must have fluency in English and French or Dutch." is unclear regarding its precise meaning: on the one hand, it can be read as requiring two languages, one of them being English and the other being either French or Dutch; on the other hand, it can be interpreted as either requiring Dutch or requiring both English and French.

3.3 Model Architecture

In order to detect the ambiguities and contradictions referred to in the previous section, we developed a four-step model that combines NLP, ML and rules, as summarized in Figure 1.

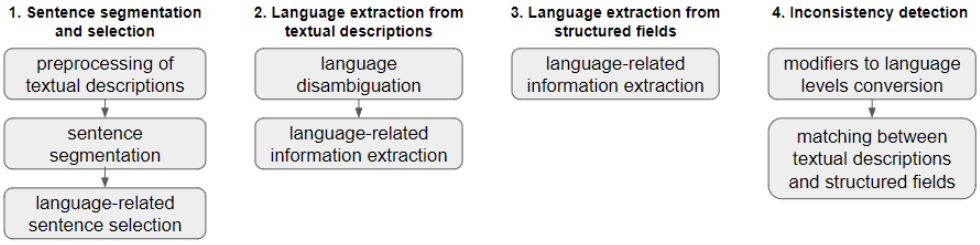

Figure 1 Overview of the Inconsistency Detection Model for language-related requirements.

The model starts by preprocessing the jobs' textual description and identifying the sentences related to language requirements, as detailed in Section 4). At step 2, it extracts the languages mentioned in these sentences, the modifiers used to describe the language levels and the remaining language attributes described in Section 3.1, including the possible existence of ambiguities (Section 5). At step 3, the model extracts language-related information from the structured fields. Finally, at step 4, the model compares the information extracted from the textual descriptions with the information from the structured fields to detect any contradictions that may exist (Section 6).

4 Sentence Segmentation and Selection

4.1 Preprocessing of Textual Descriptions

The first step of the model is the preprocessing of the textual descriptions of job postings. As these descriptions are inserted in our web-based recruitment tool, the preprocessing step consists of common NLP tasks such as the removal of URLs, symbols, HTML tags and HTML entities, and the trimming of white spaces.

4.2 Sentence Segmentation

Once cleaned, textual descriptions are segmented into sentences using a third-party NLP tool [3]. Example 2 illustrates the segmentation of the textual description of Example 1 into four distinct sentences. In this phase, some of the resulting sentences may occasionally consist of punctuation marks or single characters, which are removed by our model as they are meaningless. Also, we consider that semi-colons do not break up a sentence.

▶ **Example 2.** Result of the segmentation of the textual description of Example 1:

1. The applicant must have relevant experience in the domain and, at least, a masters in biology, biochemistry, materials science or related areas.
2. S/he must be autonomous and very organized.
3. Experience in statistics is valued.
4. We expect good knowledge of English and similar knowledge of either French or Portuguese; German is considered an asset.

4.3 Language-related Sentence Selection

After segmentation, the model identifies and selects the sentences that are related with language requirements. We tackle this task as a ML binary classification problem. For that, we trained a Random Forest model [5] using a corpus of real job sentences written in English, where each one of these sentences was labeled by us as either 1 (contains at least one mention to a language requirement) or 0 (does not contain any language requirement). The choice of the ML approach at this stage was due to the fact that it generalizes better to a more abrangent model of detection of inconsistencies where there is the need to isolate sentences related to different concepts, such as languages, education, and experiences. Also, we believe that ML may provide contextual benefits in the presence of ambiguous cases (e.g., "proficient in IT" makes it clear that the sentence refers to "Italian" and not to "Information Technology") or in extracting the best weights to apply to different types of words that are relevant to identify a language (an example of such words is "excellent", which appears frequently associated with a language but that can be used in a different context).

Each sentence of the corpus was lowercase and stemmed and, then, represented as a vector of a fixed size, corresponding to the number of features used in the model. These features are words that occur more frequently in language-related sentences, such as language names and language codes, language modifiers such as "fluent", "proficient" and "native", language-related verbs such as "speak", "write" and "read", and language related terms like "level" and "language". The resulting vectorized sentences accounted for the existence (1) or absence (0) of each one of the features in the sentences (one-hot encoding), as illustrated in Figure 2 for sentences "We expect good knowledge of English language" and "Experience in statistics is valued". The vectors shown in this figure are simplified for the sake of clarity.

	english	french	portugues	en	fr	knowledg	languag	...
we expect good knowledg of english languag	1	0	0	0	0	1	0	...
experienc in statist is valu	0	0	0	0	0	0	0	...

Figure 2 Simplified example of vectorized sentences.

Finally, we split the data into training and test sets and trained a Random Forest model to learn the task of classifying sentences as either containing at least one mention to a language requirement or not containing any language reference.

4.4 Validation

In this section, we evaluate our model to select language-related sentences from a corpus of textual descriptions of job postings. It does not include the evaluation of the accuracy of the sentence segmentation, as this segmentation was mostly done by a third-party tool [3].

We built up a set of 5149 sentences that resulted from the segmentation of 566 textual description of job requirements written in English, from our recruitment corpus. 574 of these sentences contained at least one mention to a language requirement. We split this dataset into train and test sets using an approximate ratio of 8:2. Table 2 summarizes the distributions of sentences, jobs and positive cases (sentences mentioning at least one language requirement) in both sets. The distribution of positive cases followed the train/test ratio.

Table 2 Train and test instances for step 1 of the model.

	sentences	**jobs**	**positive cases**
train	4267	478	490
test	882	88	84

We evaluated this model in the test set. The results have shown 99.21% of accuracy, 95.24% of recall and 95.81% in the f1-score. Because the correct identification of language-related sentences is the focus of this step of our model, we paid particular attention to the *recall* metric. We performed an error analysis on the four cases where the model failed to detect a reference to a language. Three of these cases were similar to the sentence "other languages are an asset" and have no immediate impact on the detection of inconsistencies. The fourth case is "Luxembourgish is an asset", and it is due to the case that "Luxembourgish" appears frequently in our corpus associated to other concepts than languages (e.g., "Luxembourgish law", "Luxembourgish offices"). We must address this issue in future work.

5 Language Extraction from Textual Descriptions

The second step of our model applies to the sentences identified as mentioning at least one language requirement in the previous step. For each one of these sentences, the model extracts the language names appearing in the sentence, their corresponding language-level modifiers, should they exist (e.g., "fluent", "native"), whether the languages are optional or non-optional, and whether non-optional languages are considered alternative or not. The model also derives the number of required languages stated in each sentence and whether the sentence is ambiguous or not. An illustrative example is given in Table 3.

Table 3 Example of information extracted in Step 2 for sentence "Excellent English skills, both written and verbal, and a fluent knowledge of French and Dutch".

language code	**modifier**	**optional**	**alternative**
en	excellent	no	no
fr	fluent	no	no
nl	fluent	no	no
Required languages: 3, Ambiguity: no			

5.1 Language disambiguation

The extraction algorithm starts with a *language disambiguation phase*, where sentences are searched for specific terms that could denote either a language code or some other unrelated concept. As an example, the term "HR" is commonly used in the recruitment domain as an acronym for "human resources" but it can also be the two-letters code for the Croatian language. Similarly, the term "IT" can be an acronym for "information technology" or the language code for Italian, and the term "PL" can be the acronym for "programming language" or it can be the language code for Polish. In order to cope with these situations, we built a controlled vocabulary of words related to these possible other meanings of specific language codes. For instance, to cope with the "PL" term, we added to this vocabulary words related with programming languages, such as "R", "Python", "Java", and "C". Then, in the presence of one such ambiguous language code in the sentence under analysis, the algorithm searches for one or more occurrences of the related terms in the vocabulary, and if occurrences are found the algorithm drops the term from the list of possible languages.

5.2 Modifier detection

After the extraction of languages, the algorithm proceeds with the detection of the language level modifiers associated with each one of the languages identified in a sentence.

We verified that the standalone use of dependency parsing or part-of-speech techniques to automatically extract these language level modifiers led to different types of errors, mostly when there are different languages and modifiers in the same sentence or when sentences are verbless (e.g., "Fluent in French and English proficiency"), which are common in the description of language requirements. On the other hand, a thorough analysis of the positioning of modifiers in hundreds of language-related sentences of real job postings in our corpus allowed us to identify different patterns of distancing and positioning between languages and modifiers that could be converted into syntactic rules to extract these language modifiers, or level indicators. As an example, Figure 3 illustrates the positioning of the modifier "excellent" relative to language "English" and the positioning of modifier "fluent" that affects both languages "French" and "Dutch".

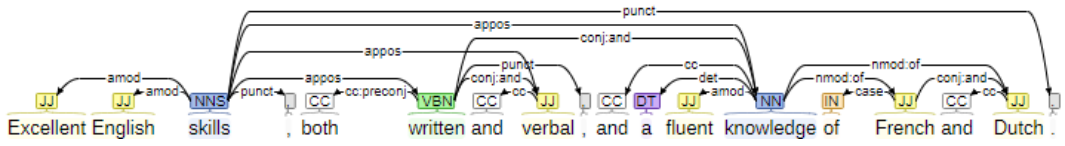

Figure 3 Use of dependency parsing to measure the distances between languages and modifiers in sentence "Excellent English skills, both written and verbal, and a fluent knowledge of French and Dutch". [3]

Next, we present some of the patterns of modifiers' identification, positioning and distancing to the correspondent languages that were more common in our corpus and that dictated our rule-based model.

1. *The number of language modifiers is relatively small.* We verified that the modifiers used by different recruiters in different roles and domains do not vary substantially. Therefore, we built a list of possible modifiers that the algorithm should look at, starting with a vocabulary of modifiers that we found in our recruitment corpus (e.g., "fluent",

[3] Figure generated using CoreNLP tool [11].

"proficient", "solid") and manually extending it with relevant synonyms with the help of NLP tools such as WordNet [7] and qdap [15]. This list was later validated by our recruitment experts.

2. *Some modifiers comes immediately before the language*, as in "We require good English".
3. *Some modifiers come before the language but not immediately.* An example of this pattern is "Fluent in writing and reading in Norwegian". In this case, we verified that the words between the modifier and the language could be removed without impairing the correct identification of the modifier, resulting in a sentence matching Pattern 2. We created a list with such *neutral* words (e.g., "writing", "reading", "skills" and "presentation"), augmented with synonyms of these obtained from WordNet and qdap.
4. *Some modifiers come immediately after the language*, as in "Speak English fluently".
5. *Some languages have more than one modifier.* An example of this is "Good knowledge of English", where both "good" and "knowledge" are on the list of possible modifiers. We identified specific words, such as "knowledge", "communication", and "skills", which are ignored by our extraction model when they appear adjacent to other modifiers.
6. *Some modifiers apply to more than one language separated by connective "and".* An example of this pattern is "Workable English and French".
7. *Some languages do not have a modifier*, as in "Swedish is mandatory".
8. *Some sentences have duplicated languages but just one possible modifier.* One example is "Native level of spoken and written English and Spanish workable knowledge (English is of utmost importance)". In this sentence, "English" is duplicated, but neither one of the terms in "is of utmost importance" appear in our list of possible modifiers. Therefore, the modifier associated to English extracted by our model is "native".

5.3 Detection of optional, alternative and required languages

After language and modifier extraction, the algorithm checks which languages are optional/ non-optional, and which non-optional languages are alternatives. We found the following patterns:

1. *The description of optional languages is often accompanied by specific terms*, as in "French and English; Spanish and Portuguese being an advantage", where the term "advantage" makes obvious that Spanish and Portuguese are optional languages. In reality, the analysis of our corpus has shown that a simple search for specific words (e.g., "asset", "advantage", "plus") proved to be an efficient approach, so we built a list with such *optional* words extended by synonyms.
2. *The description of non-optional languages is often accompanied by specific terms*, such as "mandatory", "required", "must", "compulsory" and "essential", as in "Swedish is mandatory". We built a list of such *non-optional* words, extended with synonyms, and consider a language as non-optional if it is associated to any term of this list.
3. *Alternative languages are often associated with specific terminology (e.g., "either", "as well as", "combined with", and "together with"), connectives ("or") and related indicators ("/").* Examples of these are "You should master French or English" and "English and either French or German". We use this expressions to extract alternative languages.

5.4 Ambiguity and number of required languages detection

Some language-related sentences are ambiguous and the extraction algorithm should not only signal these ambiguities but also take decisions in the presence of these ambiguities. As an example, the sentence "Profound written knowledge of English, Dutch and/or German

are required, Luxembourgish is a plus" raises a certain level of ambiguity concerning the *minimum* number of required languages, which could be either one (English or Dutch or German) or two (English and either Dutch or German). As a different example, the sentence "You must be English/French fluent" raises an ambiguity concerning the meaning of "/" in this sentence, which could indicate that the recruiter wants the applicant to know English *or* French or English *and* French. However, if we consider the sentence "You must be English/French bilingual", the term "bilingual" helps to disambiguate the sentence. As a final example, the algorithm understands the sentence "Fluency in English and Spanish or French" as corresponding to two minimum required languages, all non-optional, and two alternative languages (Spanish and French), but it also raises an ambiguity as this sentence allows for different interpretations. When all information about the optional/non-optional and alternative languages is extracted, the number of *minimum* required languages described in a sentence is computed as the sum of all non-optional languages, to which a unit is added if in the presence of, at least, one alternative language.

5.5 Validation

To evaluate the extraction algorithm, we annotated a set of entries corresponding to 733 language requirements (715 of these specifying a specific language) in 371 language-related sentences, from 302 textual description of job postings. More concretely, for each language mentioned in each sentence (an *entry*), we annotated the language code, its modifier, whether the language was optional or not and whether the language was alternative or not. At the sentence level, we annotated the number of required languages mentioned in the sentence and whether or not the sentence was ambiguous. Table 4 further characterizes this dataset, presenting the frequency of the labels for each one of these components. As a note, "–" values are related to sentences that mention a language requirement without specifying a language name, and "others (n)" aggregate the frequencies of n infrequent values.

Table 4 Frequency table for the labels used in step 2.

modifier	fluent: 31%, (no modifier): 24%, fluency: 10%, excellent: 9%, participate: 4%, good: 3%, –: 2%, basic: 1%, communication: 1%, knowledge: 1%, very good: 1%, outstanding: 1%, proficiency: 1%, strong: 1%, others (22): 10%
language	EN: 42%, FR: 29%, DE: 10%, LB: 4%, NL: 3%, –: 2%, others (18): 10%
optional	no: 82%, yes: 15%, –: 2%
alternative	no: 89%, yes: 11%
required	2: 63%, 1: 30%, 3: 5%, 4: 1%, 0: 1%
ambiguity	no: 94%, yes: 6%

We split this dataset into training and test sets using a 7:3 ratio, where the training data was used to develop the rules and the test set to validate the model. The number of entries, sentences and jobs for the training and test sets is shown in Table 5. We run our rule-based extraction algorithm on the test set, for each one of the extraction phases. Errors in one phase do not propagate into the following one in this test settlement, except for error in the extraction of optional languages, which propagates into the extraction of the *minimum* required languages of the job posting and into the extraction of ambiguities.

We obtained 100% of accuracy in the extraction of the language names. Table 6 presents the results obtained for the extraction of the optional, alternative and ambiguity features, and Table 7 presents the errors per class and the accuracy for the extraction of modifiers

Table 5 Train and test instances for step 2.

	entries	sentences	jobs
train	529	262	216
test	204	109	86

and for the *minimum* number of required languages per sentence.

Table 6 Results for step 2 for binary classes.

	accuracy	recall	f1-score
Optional/Non-Optional	94.9%	98.04%	96.77%
Alternative/Non-Alternative	98.09%	100%	98.96%
Ambiguity	94.18%	83.33%	66.67%

Table 7 Accuracy and number of errors for modifiers and number of required languages.

label	#errors	accuracy
modifier	fluent: 4, fluency: 4, good: 1, knowledge: 1, others: 0	94.90%
required languages	0: –, 1: 0, 2: 1, 3: 1, 4: 0	97.67%

From the results, we associate the perfect accuracy of the model in extracting the language names to the fact that recruiters tend to write languages and language codes correctly, however, we intend to reinforce typos checking in a future version of the model.

An important result to look for is the ability of the extraction model to distinguish between optional and non-optional languages, because errors on this phase propagate to posterior phases of the model (in a production environment), such as the detection of alternative languages and the computation of the *minimum* number of required languages per sentence. Therefore, we were looking for high values of recall, and our model was able to detect the non-optional languages with a value of recall of 98.04%. An error analysis on unsuccessful cases have shown that some of them are hard to detect because of structural malformations of the sentence – e.g., "Portuguese and Spanish (of advantage)". In some other cases, we verified that an additional rule will be needed to be added to our set of rules. Finally, we observed that the accuracy of the *minimum* number of required languages per job posting was 91.86%, reflecting the propagation of errors from the optional/non-optional phase.

The results have also shown high performance in the extraction of alternative languages, with 98.09% of accuracy and 100% of recall. Concerning the extraction of modifiers, our model achieved an accuracy of 94.90%. The error analysis have shown that the errors were mainly associated to the existence of multiple modifiers assigned to the same language and to the existence of large distances between languages and modifiers. We intend to address the tuning of our neutral-words skipping process, as well as the possibility to add another rule to fix these errors, in future work. Finally, the algorithm for the extraction of ambiguities achieved an accuracy of 94.18% and 83.33% of recall, with one single false negative in sentence "Portuguese and Spanish (of advantage)". The majority of false positives propagated from the detection of optional languages phase. It also become evident to us that we have to add more ambiguous sentences to our dataset in a future evaluation of our inconsistency model.

6 Inconsistency Detection

The third step of our inconsistency detector model is the detection of contradictions in language requirements, by comparing the information extracted from textual descriptions to the information specified in the structured fields, and by assigning a specific type of inconsistency (cf. Section 3.2) to the inconsistent sentences.

6.1 Conversion of modifiers to language levels

This step starts with the conversion of the language modifiers extracted by the model into language levels, as it is the way this information is present in structured fields. As an example, "native [English]" must be converted into level C2, and "basic knowledge [of French]" must be converted into A2. In order to make these conversions, we asked our recruitment experts to categorize all modifiers of our extended modifiers list into categories of similar semantics, using the technique of card sorting [21]. At the end of this process, our experts assigned a language level (from A1 to C2) to each group of similar modifiers, and indicated the *minimum* and *maximum* levels that could be associated to a given language level without being considered *contradictory* to that language level. As an example, the modifiers "fluent", "proficient", "perfect", and "flawless" were assigned a *minimum* level of C1 and a *maximum* level of C2. Therefore, if a given sentence mentions "Perfect Spanish"and the associated level in structured fields is B2, this would raise a *lexical* contradiction. As card sorting was done separately for each one of our experts, at the end of this process they met to discuss the modifiers that were grouped differently or for which the level assignment was different and a decision was made by consensus. The result of this process is a validated dictionary of modifiers-language levels correspondences that our model uses in this initial stage.

6.2 Matching between textual and structured fields

The model proceeds by matching each item of the extracted information with the corresponding item of the structured fields, using a set of rules that allow the model to raise zero or more contradictions. Next, we present a set of examples illustrating the model's rules.

▶ **Example 3. The applicant must be fluent in English and in French**

Structured Fields				Textual Description			
language	**level**	**optional**	**altern.**	**language**	**modifier**	**optional**	**altern.**
en	B2	no	no	en	fluent	no	no
				fr	fluent	no	no
Required languages: 1				Required languages: 2, Ambiguity: no			

Here, the textual description refers to a language that is not defined in the structured fields (French), which also implies that the number of required languages does not match, and the model raises the *Language-not-specified* and *Numerical* contradictions.

▶ **Example 4. Fluent in Norwegian and/or Italian and English**

Structured Fields				Textual Description			
language	**level**	**optional**	**altern.**	**language**	**modifier**	**optional**	**altern.**
no	B2	no	yes	no	fluent	no	yes
it	A2	no	yes	it	fluent	no	yes
en	B2	no	no	en	fluent	no	no
Required languages: 2				Required languages: 2, Ambiguity: yes			

This is one typical example that raises an *Ambiguity* alert. In this case, the model assumes that there are at least two non-optional languages, one being English and the other being either Norwegian or Italian, and we verify that the logic associated with the precedence of the connectives ("and", "or") and with the use of "/" is accompanied by the *world knowledge* that English is an Universal Language. This would not the case in other situations, so the use world knowledge in inconsistencies detection should be addressed in future work. Finally, the model would raise a *Lexical Contradiction* because the textual description mentions that the applicant must be fluent in one of the alternative required languages (Norwegian or Italian) and the language level associated to Italian is A2, which corresponds to a basic level, below the *minimum* language level associated with modifier "fluent" by our recruitment experts.

▶ **Example 5. English is mandatory**

Structured Fields				Textual Description			
language	**level**	**optional**	**altern.**	**language**	**modifier**	**optional**	**altern.**
en	A1	no	no	en		no	no
Required languages: 1				Required languages: 1, Ambiguity: no			

In this example, the model does not raise any contradiction or ambiguity. As the model does not extract any modifier for English, it would not check for lexical contradictions.

▶ **Example 6. Work knowledge of German or Dutch**

Structured Fields				Textual Description			
language	**level**	**optional**	**altern.**	**language**	**modifier**	**optional**	**altern.**
de	B2	no	no	de	knowledge	no	yes
nl	B2	no	no	nl	knowledge	no	yes
Required languages: 2				Required languages: 1, Ambiguity: no			

This is another typical example appearing in our corpus. The textual description asks for one required language but the structured fields defines two required languages, ignoring that both German and Dutch should be defined as alternative languages. In this case, our model raises an *Alternative-languages contradiction* and a *Numerical contradiction*.

6.3 Validation

We validated this step of our model with a dataset composed of 353 language-related sentences from 302 textual descriptions of job postings, corresponding to 715 instances of languages (*entries*). Table 8 presents the frequency table for each type of contradiction. The number of entries, sentences and jobs for the training and test sets is shown in Table 9.

Table 8 Frequency of each type of contradiction in the dataset of step 4.

Lang-not-spec	Lang-not-req	Lang-not-opt	Lexical	Mandatory	Numeric
5%	3%	2%	10%	3%	15%

We run our algorithm in the test set and it achieved an accuracy of 100% in detecting contradictions of types *Language-not-specified*, *Language-not-required*, *Language-not-optional*, *Alternative-language* and *Numerical* (per posting). Regarding the detection of *Lexical* contradictions, the model was accurate 98.98% of the times, reaching a recall of 91.30% and a f1-score of 95.45%. From error analysis, we verified that the errors in this phase were related to cases where the recruiters were more exigent in the language requirements as specified

Table 9 Train and test instances for step 3 of the model.

	entries	sentences	jobs
train	519	252	216
test	196	101	86

in structured fields than in the textual descriptions of job postings, as when they asked for "Good knowledge of English"and then asked for C1 levels in structured fields. Our team of experts concluded that this type of contradiction is not as severe as the one occurring in the opposite direction (e.g., asking for "Native English"and then defining B2 levels).

7 Conclusions

This paper presented an approach to detect contradictions and ambiguities in the description of language requirements in job descriptions written in English. We focused the content of the paper in two essential components. First, we provided and analyzed a set of examples of common language-related sentences containing at least one ambiguity or that are contradictory when compared to the language requirements specified in the job's structured fields. Then, we proposed a terminology for the description of inconsistencies in language requirements, composed of six types of inconsistencies and one ambiguity. This proposal resulted from the thorough analysis of our corpus of job descriptions from hundreds of distinct job roles published by different recruiters from several organizations, in different countries and industries.

Second, we proposed a four-step NLP-based model to detect these inconsistencies from job descriptions. This model uses machine learning to extract the language-related sentences (step 1) and a set of comprehensive rules to extract relevant information from these sentences (step 2) and to detect the inconsistencies (step 4). We have shown that even with a restricted set of rules the model achieved high performance in each one of the steps. Moreover, this model will serve as a baseline to further improvements, which can include the use of a machine learning approach to extract the mentioning languages and their requirements from sentences.

As future work, we intend to tune our existing rules to fix error cases detected in error analysis and to provide a more sophisticated approach to detect alternative languages. We also intend to enrich the NLP preprocessing with typo checking. Although we are convicted that our annotated dataset of job postings covers the majority of possible sentences describing language requirements, we still intend to extend it (namely, re-enforcing the number of ambiguous sentences), in order to evaluate how the model would scale to different types of sentences. Finally, we believe that this approach adapts well to sentences written in other languages, such as French and Portuguese, so we intend to adapt it to these languages.

References

1 Edward Tristram Albert. Ai in talent acquisition: a review of ai-applications used in recruitment and selection. *Strategic HR Review*, 2019.

2 Harald Alvestrand. Tags for the identification of languages. *RFC 1766, March*, 1995.

3 Kenneth Benoit, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. quanteda: An r package for the quantitative analysis of textual data. *Journal of Open Source Software*, 3(30):774, 2018.

4 Daniel M Berry, Erik Kamsties, and Michael M Krieger. From contract drafting to software specification: Linguistic sources of ambiguity. *A Handbook*, 2003.

5 Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
6 Valentina Dragos. Detection of contradictions by relation matching and uncertainty assessment. *Procedia Computer Science*, 112:71–80, 2017.
7 Christiane Fellbaum. Wordnet. *The encyclopedia of applied linguistics*, 2012.
8 Alessio Ferrari, Beatrice Donati, and Stefania Gnesi. Detecting domain-specific ambiguities: an nlp approach based on wikipedia crawling and word embeddings. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, pages 393–399. IEEE, 2017.
9 Benedikt Gleich, Oliver Creighton, and Leonid Kof. Ambiguity detection: Towards a tool explaining ambiguity sources. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 218–232. Springer, 2010.
10 Luyang Li, Bing Qin, and Ting Liu. Contradiction detection with contradiction-specific word embedding. *Algorithms*, 10(2):59, 2017.
11 Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.
12 Marie-Catherine Marneffe, Anna N Rafferty, and Christopher D Manning. Finding contradictions in text. *Proceedings of ACL-08: HLT*, pages 1039–1047, 2008.
13 Council of Europe. Council for Cultural Co-operation. Education Committee. Modern Languages Division. *Common European Framework of Reference for Languages: learning, teaching, assessment*. Cambridge University Press, 2001.
14 Minh Quang Nhat Pham, Minh Le Nguyen, and Akira Shimazu. Using shallow semantic parsing and relation extraction for finding contradiction in text. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1017–1021, 2013.
15 Tyler W Rinker. qdap: Quantitative discourse analysis package. *University at Buffalo/SUNY*, 2013.
16 Allan Berrocal Rojas and Gabriela Barrantes Sliesarieva. Automated detection of language issues affecting accuracy, ambiguity and verifiability in software requirements written in natural language. In *Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas*, pages 100–108. Association for Computational Linguistics, 2010.
17 Benedetta Rosadini, Alessio Ferrari, Gloria Gori, Alessandro Fantechi, Stefania Gnesi, Iacopo Trotta, and Stefano Bacherini. Using nlp to detect requirements defects: an industrial experience in the railway domain. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 344–360. Springer, 2017.
18 Ali Olow Jim'ale Sabriye and Wan Mohd Nazmee Wan Zainon. A framework for detecting ambiguity in software requirement specification. In *Information Technology (ICIT), 2017 8th International Conference on*, pages 209–213. IEEE, 2017.
19 Karin Kipper Schuler. Verbnet: A broad-coverage, comprehensive verb lexicon. *Ph. D. Thesis, University of Pennsylvania*, 2005.
20 Adam Sennet. Ambiguity. *Stanford Encyclopedia of Philosophy*, 2011.
21 Donna Spencer and Todd Warfel. Card sorting: a definitive guide. *Boxes and arrows*, 2:1–23, 2004.