Technical Term Extraction Using Measures of Neology

CHRISTOPHER NORMAN
Technical Term Extraction Using Measures of Neology
Facktermsdetektering medelst neologiska kriteria

Christopher Norman
chnor@kth.se

supervised by  Jussi Karlgren (Royal Institute of Technology)
Akiko Aizawa (University of Tokyo)

examinated by  Jens Lagergren (Royal Institute of Technology)

January 26, 2016
This report details the work conducted as part of my graduation thesis work at the Royal Institute of Technology, and was carried out during my exchange at the University of Tokyo, at the Aizawa laboratory at the National Institute of Informatics, Tokyo.

First I wish to thank my supervisors, my supervisor at the University of Tokyo, Akiko Aizawa, and my supervisor at the Royal Institute of Technology, Jussi Karlgren, for all the support and advice you have given me while I have been writing this thesis.

I also wish to thank Aldo Lipani, Pontus Stenetorp, and everyone in the Aizawa laboratory for gracefully provided me with suggestions for improvement and feedback, as well as for being awesome in general.

My most heartfelt thanks go to Noriko Katsu at nii, who helped me immensely during my stay in the Aizawa Laboratory. Without your patience and expertise I would have spent half my time just finding out where to send all the paperwork!

Parts of this thesis has been published in the 2015 Beijing ACL workshop on Novel Computational Approaches to Keyphrase Extraction under the name Technical Term Extraction Using Measures of Neology. Explicitly, most of the material therein has been grafted into the various sections of this thesis, rewritten and adapted to fit the format. Four anonymous reviewers of the workshop paper as well as members of the audience at the workshop provided useful suggestions for improvements and further work. Some of the suggestions were included in the workshop paper, some of them were included in this thesis, and still others were regrettably left for future work. Those suggestions that have made it into this thesis have improved its quality appreciably, for which the author is incommensurably grateful.

And last, but certainly not least, this project would not have been possible if Panot Chaimongkol had not painstakingly annotated the SemEval-2010 dataset, for which I am most humbly grateful.
Abstract

This study aims to show that frequency of occurrence over time for technical terms differs from general language terms in the sense that technical terms are strongly biased to be recent occurrences, and that this difference can be exploited for the automatic identification and extraction of technical terms from text. To this end, we propose two features extracted from temporally labelled datasets designed to capture surface level \( n \)-gram neology. The analysis shows that these features, calculated over consecutive bigrams, are highly indicative of technical terms, which suggests that technical terms are strongly biased to be surface level neologisms. Finally, we implement a technical term extractor using the proposed features and compare its performance against a number of baselines.

Sammanfattning

Detta arbete ämnar visa att den tidsberoende frekvensen för facktermer skiljer sig från motsvarande frekvens för termer i vardagligt språk, i det avseendet att facktermer med hög sannolikhet är lingvistiska nybildningar, samt att denna iakttagelse kan nyttjas i syfte att automatiskt identifiera och extrahera facktermer i löptext. I detta syfte introducerar vi två särdrag extraherade från kronologiskt an- noterade datamängder avsedda att fånga nybildningar av förekommande \( n \)-gram. Analysen visar att dessa särdrag, beräknade över konsekutiva bigram, är starkt indikativa för facktermer, vilket antyder att facktermer har en starkt tendens att vara nybildningar. Slutligvis implementerar vi en facktermextraktor baserad på dessa särdrag och jämför dess prestanda med ett antal referenssärdrag.
# CONTENTS

1 Introduction 1

2 Background 3
   2.1 Basic Definitions 3
   2.2 Linguistic Features of Technical Terms 5
   2.3 Polysemy 8
   2.4 Terminological Neology and Terminology Change 9
   2.5 Applications 12
   2.6 Term Extraction 13
   2.7 Keyphrase extraction 15
   2.8 The Google Ngrams Datasets 20

3 Method 23
   3.1 Indexing Backend 23
   3.2 Problem Definition 24
   3.3 Neological Features 26
   3.4 Baselines 29

4 Analysis 33
   4.1 Choice of n-gram order 33
   4.2 Class separation 33
   4.3 Relevance and usefulness of individual features 34
   4.4 Named entities 41

5 Results 45
   5.1 Intrinsic Evaluation 45
   5.2 Extrinsic Evaluation 49

6 Conclusions 55

Bibliography 57
Current approaches to technical term extraction generally rely on term frequency statistics. Such approaches have proven to be quite effective, but they do come with an unfortunate side effect: for such statistics to be useful we need datasets that are large enough, containing documents with just the right size – otherwise the statistics will not be meaningful. If the dataset contains too few documents then interdocument statistics do not tell us anything. Similarly, if the documents themselves are too small then the interdocument statistics do not tell us anything. We might also encounter problems if the documents are too large, because then the statistics might be drowned out by the noise in the data (Hasan and Ng, 2014). Preferably, we would like to have methods that can be used for documents of any size, and collections with any number of document, even collections containing only a single document, or documents consisting of a single sentence.

In order to create technical term extraction methods that do not possess these shortcomings, we would need to turn away from the commonly used document statistics based approach and try to find something entirely different.

To that end, this study proposes using measures of neologism as an indicator of technical terms. Since neologism is in this study defined in terms of how recently a given term was coined and adopted for any $n$-gram term, all we would require of the document is that we can lexicalize it into $n$-gram terms. No further syntactic analysis would be necessary and the approach should thus for the most part be language agnostic. Furthermore, the approach would be independent of the size of the document – it should work equally well when the document is a single sentence as when the document is a whole book. We should even be able to use such an approach in cases where we only know the candidate technical term and nothing else, although in that case we might not be able to use complementary lexical analysis such as POS tagging.

Of course, using neologism as an indicator of technical terms requires other assumptions about what constitutes technical terms, in particular the assumption that terminology tend to be relatively recent coinage. This appears to be true in science and technology, but implausible in e.g. law where much of the terminology is several centuries old (Lemmens, 2011). Furthermore, neology on its own cannot be a complete characterization of terminology since all neologisms are of course not technical terms. However, despite these drawbacks, the benefits of a characterization that is independent of document size and data collection size still makes this line of inquiry promising. At the moment of writing, and to the best of the authors knowledge, this is still left unexamined.

---

1 Other definitions, focusing on qualitative rather than quantitative measures, are common, especially in linguistics.
Technical terms might seem like a straightforward concept – most of us can probably recognize technical terms on sight – but it turns out that exact definitions are in fact highly contentious (Kageura et al., 1999) and vary significantly from author to author (Manning and Schütze, 2000). Some authors in technical term extraction have consequently taken a laissez-faire approach to the actual definition and refrain altogether from explicitly defining technical terms and instead simply define technical terms to be those terms that occur in technical dictionaries (Justeson and Katz, 1995).

Depending on our particular needs and interests we might in fact expect to end up with different definitions of what constitutes terminology. Do we want a small number of terms as a description what a document is about? This question is central to keyphrase extraction. Do we want to find all technical terms in the document that are central to the results of the paper? We might take this stance if our goal is information extraction. Do we want to find all technical terms in the document, even those that are not central to the paper? This last stance might be natural if our goal is machine translation, in which case we would want to ascertain that terms are translated as a single unit and not word by word.

Rather than specifying absolute criteria we have to resort to general guidelines and observations of general characteristics of technical terms. In this endeavor we will be helped by a large body of literature from linguistics and terminology that is large unquoted by the computer science oriented literature.¹

In this section we will also lay the groundwork for the remainder of this study. We will explicitly define most of the terms used, in order to clear up any potential ambiguities that the reader might run into.

Several of the sections deal with theoretical, linguistic aspects of terms and terminology. These might seem overly theoretical and of little interest to the main points that the study seeks to address, but most of the issues described hide subtle difficulties which can easily trip the unvary. Formal definitions are similarly introduced for most of the terminology used in the study. Many of the definitions might seem obvious and thus unwarranted, but in almost all cases terminology differs considerably between authors and, where the terms occur in general language, only obliquely have the same meaning as the general language terms.

Finally, we will also give an overview of the relevant literature as well as algorithms and approaches currently used for extracting technical terms.

2.1 Basic Definitions

A word may be defined as units of “isolated meaning” (Sapir, 1921), but conventionally we may simply define words orthographically as those lexical units that are separated by spaces (Bauer, 1983). From a linguistic point of view the first definition is probably more interesting, but from a purely descriptive point of view, which is probably the one most often encountered in NLP, the second definition is generally adequate. For our purposes here we can probably satisfy ourselves to use whatever intuitive definition of words the reader possesses, since nuances in the definition are unlikely to cause any confusion. We simply hasten to point out that

¹Note for instance that recent application papers refer to either Justeson and Katz (1995) or to Hulth (2005b) for the observation that terms tend to be noun phrases. This has been known in the field of terminology since at least 1979 (Rey, 1979).
2. Background

despite the seeming simplicity of the concept, which has existed since antiquity (Rey, 1979), it is actually quite difficult to define. Consider for instance the difference between compound words in e.g. English and German (e.g. wind energy production vs Windenergieproduktion).

Other difficult edge cases include enclitics such as the n’t in don’t or the postpositional enclitic que in Latin (e.g. Senatus Populusque Romanus), and separable verbs in Chinese. In practice we often have to make careful decisions about how to deal with these, but as long as our concern is purely theoretical we can omit such decisions.

We also make a distinction between a subject field, which we use in an absolute sense, and a domain, which we use in a relative sense. Thus we talk about biology as a subject field when we refer to it on its own, but as the domain of biological articles or scientists.

Central to the field of terminology are concepts which are abstract representations of objects within a subject field (ISO 704:2009, 2009, Kageura, 2002; Rey, 1979; Sager et al., 1980). Terms are then conventionally defined, using this definition, as those lexical units that denote specific concepts in the subject field. Traditionally, the practitioners in the field of terminology has taken the seemingly platoonic stance that concepts are the primary, or even the only point of consideration, separate from and prior to language (Kageura, 2002; Rey, 1979; Sager et al., 1980; Temmerman, 2000). This view may in part be explained by the fact that the field of terminology originated outside of linguistics (Rey, 1979; Temmerman, 2000), but such a view, in which linguistic considerations such as morphology or syntax are irrelevant, would make it impossible for us to compare linguistically equivalent variants of terms, or to use variation as an indication of hyponymy relations (e.g. Daille, 2005). In this study we will not focus on the conceptual aspects of terminology, except by pointing out that the denotation of a specific domain concept is generally considered a prerequisite when deciding whether a term candidate should be classified as a term or not.

The important point in this discussion is the centrality of what is variously referred to as the concept system (ISO 704:2009, 2009), conceptual structure (Kageura, 2002), taxonomy (Rey, 1979), or the ontology (Rey, 1979) of the subject field. Our main interest should lie in trying to describe the totality of concepts in the subject field, rather than in the linguistic concerns of the terms that denote its contents. That said, we should take any chance we can to use any linguistic knowledge that helps us towards this end. Regardless, the condition that terms denote specific concepts is helpful to distinguish terms from non-terms, as well as for providing arbitration in cases of potential polysemy or synonymy.

Linguistics is in general concerned with the study of Languages for General Purposes (LGP) or General Languages. By contrast, in terminological studies we are concerned with the study of terminologies, which, by definition, are only defined for specific fields of study. Consequently, again by definition, terminologies do not exist for general languages but only for the specific languages used within the individual subject fields, which we will, as is conventional, call Languages for Special Purposes (LSP) or Special Languages. Special languages are often derisively called jargon and decried as an obstacle to public understanding, presumably in the belief that practitioners of a subject field construct specialized language merely as an exercise in hermeticism (Sager et al., 1980). While some practitioners of law reportedly use jargon as an instrument of intimidation against non-practitioners (Lemmens, 2011), the main reason for the divergence between general and special languages can simply be attributed to different communicational needs (Sager et al., 1980). For instance, in general

---

2 It is interesting to note that this viewpoint is essentially the antithesis of the Sapir-Whorf hypothesis.

3 The quite opaque terms LGP and LSP have been declared obsolete in ISO 10241-2:2009. We will here generally use the simpler and more transparent forms, general and special language.
languages synonymy is generally a source of expressiveness, whereas in special languages any gain in expressiveness from synonymy would be greatly overshadowed by its negative implications as a source of confusion.

In practice all communication have some kind of subject topic, or topics, and as such it makes little sense to talk about general language communication. Rather, we define general language to be those parts of language that are common to the majority of the special languages.

In this study we will follow e.g. Rey (1979) and Sager et al. (1980) and define a vocabulary as the totality of lexical units in a general language, i.e. the lexical units observable in discourse in a wide range of subjects. Similarly, we define a terminology to be the totality of terms in an special language.

Neologisms are words or terms that are recent introductions to a language. That a term is recent, however, might mean several different things: that the surface form of the term is new, that an old lexical unit is used in a new sense, or that old term has been imported from another domain or language (Pavel, 1993; Rey, 1974; Sager et al., 1980). In this study, we define neologisms statistically, as a lexical unit where we can find some point in time before which the lexical unit was not widely used, and after which it became much more common. Of course, neologism is a relative concept, and there must exist some such point in time for all terms in any language. Consequently, we cannot discuss neologisms without specifying the timeframe we are considering. In this study, we will for the most part be looking at timeframes covering the least century or so, but the timeframe of interest will differ greatly between new disciplines like computer science and older disciplines like mathematics.

2.2 Linguistic Features of Technical Terms

There is no single definition that can adequately characterize technical terms, and that can be applied to consistently separate terms from non-terms. Rather, we have to resort to using several criteria in concert to distinguish terms from non-terms. At the end of the day, however, what is a term and what is not must be left to human judgment. The characterizations that follow are to be taken as general observations about terms, not as formal criteria.

**Characteristic 1. Termhood: The degree that a phrase denotes a concept in the domain.**

This characterization roughly parallels the orthodox definition given by the linguistically aligned fields of terminology and terminography – that technical terms are phrases that denote concepts. For the construction of terminologies by humans, it is considered central (ISO 704:2009, 2009; Rey, 1979; Sager et al., 1980).

However, note that names of particular entities are also frequently considered terms in their respective domain despite not being concepts. For instance, the South Pole, Jupiter, the Bill of Rights, lactase.

Unfortunately, this characterization is mainly intended for an audience of terminologists and domain experts, i.e. as a guide for human judgment. It does not provide us with any sensible yardstick by which we can automatically determine whether a term candidate should be considered a term or not. It is also a fairly subjective criterion. Consequently, the characterization in terms of termhood is left to human subjective evaluation by most researchers (Kageura and Umino, 1996, p. 274).

**Characteristic 2. Unithood: The amount of cohesion between the individual compositional elements of a multiword term.**
2. Background

The cohesion between the individual elements of terms is fairly straightforward to measure with standard statistics, using e.g. the dice coefficient, mutual information, or the $\chi^2$ measure, to cite just a few popular choices. An overview of some statistical measures applied to term extraction is given by Kageura and Umino (1996), and will not be repeated here. This overview is not comprehensive however, nor is any such overview ever likely to be, considering the vast number of available similarity measures and the vast number of attempts to use different subsets of them in term extractors. Comparisons between different measures are in general not interesting, since the lack of standard datasets for evaluations as well as the lack of publicly available implementations means that different implementations cannot be compared.

Terms generally have high cohesion can at least partly be explained by the observation that terms generally are not subject to formal variation in usage (Justeson and Katz, 1995).

The unithood is closely related to collocation, with the distinction that collocation is generally defined in terms of corpus statistics, whereas unithood refers to a somewhat wider concept also using qualitative criteria. For instance, terms, as well as other kinds of collocations, like idioms, phrasal verbs, and expressions, can generally not be completely understood simply by looking at its parts (Manning and Schütze, 2000, ch. 5). Often the parts that constitute the term or phrase only give a rough hint about its semantic contents. As an example, heavy smoker does not imply high weight. This point seems particularly pertinent to the morphemes in words constructed from Greek or Latin roots (Sager et al., 1980, p. 288). For instance, it is difficult to find the common semantic element of cosmonaut, cosmopolitan, and cosmetics, at least in a way that is useful for, say, term classification.

We can thus phrase the above characterization in a different way: A term is either non-compositional, i.e. it cannot be further divided into separate tokens, or a compositional lexical unit whose meaning is not completely explained by its parts.

Unithood is more readily measured using corpus statistics than is termhood. Consequently, most authors focus on the unithood aspect of terms, sometimes with the complete disregard of termhood. For instance, in the description of their term extractor, Pantel and Lin (2001) use a non-standard definition of terms which only uses unithood as a criteria, seemingly unaware of the distinction between collocations and terms. They evaluate using $n$-gram perplexity and Chinese word segmentation, evaluation that are motivated under this definition, but not under the conventional definition of terms. The main problem with this is that the top ranked ‘terms’ will, under such a definition, consist almost entirely of common $n$-grams like of the, and so forth, and so on (see for instance Manning and Schütze, 2000, ch. 5). This is not to say that these kinds of collocations are not useful, but this is not what we want here.

Characteristic 3. Specialized Usage: Technical terms are characteristic of its domain.

This may express itself in several ways.

For one, technical terms are much more likely to appear within their relevant domain than without. This follows from the definition of terms as constructions in their relevant domains. Terms sometimes share surface forms with terms in other domains, or with terms in general language, but polysemy is generally considered to be much rarer in special than in general language.

Another way in which technical terms are characteristic of their domains is that technical terms are defined in their domain to have very constrained meanings, and thus it follows that when technical terms occur in their domains, it is always in very restricted contexts. To give
one example, the technical term accuracy is defined in machine learning and related fields as the ratio between true positives and false positives. Consequently, whenever the technical term accuracy appear in a machine learning it always means exactly this, and it always occurs in the context of evaluation. By contrast, the non-technical term accuracy has a wider range of meanings – we can for instance talk about accurate information or accurate readings, where the semantically shared part is less well defined. The underlying meaning of the technical term and the non-technical term is essentially the same: being free of errors, but the non-technical term can carry this semantic payload to a much more diverse set of contexts.

Some commonly used statistical measures such as tf-idf are tangentially related to this characterization in that terms with high inverse document frequency might be terms that are only used in certain special languages. Another common statistical measure is using term frequency in the document set divided by term frequency in a general language corpus. Chai-mongkol and Aizawa (2013) use LDA clusters for term extraction, and Dredze et al. (2008) similarly use LDA and LSA for keyphrase extraction, but using domain-specificity as a characterization of terms is otherwise fairly uncommon. This may partly be explained by the observation that almost all term extraction systems are evaluated and/or applied on a single subject field, and consequently, that topic clustering would arguably have limited to no effect on the performance.

Characteristic 4. Parts of Speech: Technical terms are noun phrases

This characterization is commonly used for technical term extraction. Given the recent high performance of POS taggers and that this provides a completely automatic description of terms, it is not surprising that filtering for noun phrases has turned into a staple approach to term extraction. POS tagging is commonly used as a first filter to find noun and noun phrases in many term extractors (e.g., Frantzi et al., 2000; Justeson and Katz, 1995).

This characterization is somewhat too categorical, however, and might miss certain terms. Technical terms are indeed generally nouns or noun phrases (Hulth, 2003a; Justeson and Katz, 1995; Rey, 1979), but also, to a lesser extent, verbs or adjectives that cannot be reformulated as nouns (Rey, 1979).

On the other hand, the evidence for this is generally given by examining terminology dictionaries (Justeson and Katz, 1995). Thus we should take into account that when non-noun terminology, for instance crystallize or metamorphic, are entered into terminology dictionaries, they are generally converted into nouns, i.e. crystallization and metamorphic rock (Rey, 1979). Thus if we want to discover technical terms in running text, then the knowledge that technical terms are nouns in terminology dictionaries becomes less relevant.

Characteristic 5. Neology: A technical term is relatively recently formed, compared to most terms in general language.

This is an additional characterization that we propose in this study that has previously been suggested in linguistics-oriented studies (e.g. Rey, 1979; Sager et al., 1980), but does not appear to have been used as a characterization in any term extractor to date, nor even mentioned in computational linguistics-oriented studies. One possible explanation for this might

Note that the non-technical term accuracy can well appear in a machine learning paper, although such usage might well be confusing, and might thus be avoided by careful authors.

Note however that the performance is generally evaluated on general language. It is an open question whether the performance is generalizable to any given special language. For instance, consider that the performance drops appreciably when standard POS taggers are applied on historical texts (Lin et al., 2012).
be that detecting neology requires the existence of a large temporally labelled corpus, and consequently would have been difficult to perform before the publication of such datasets. To the best of the author’s knowledge, the only dataset that contains documents uniformly sampled over a period of several centuries is the Google Ngram datasets, which was only released in 2006. Since the topic of neology as a characterization of technical terms is central to this study, we will not investigate it here, but will instead afford it a separate section (Section 2.4).

We should be careful to note the difference between a statement and its converse. Here we are assuming that technical terms tend to be neologisms. However, the converse, the assumption that neologisms tend to be technical terms, is false in the general case. It is, after all, possible to construct counterexamples such as *dissing* or *walking the walk*, which are neologisms but not technical terms. This is unfortunate because it is the validity of this converse statement that we are relying on to extract technical terms using neology as a characterization. That said, we would encounter the exact same problem for any other characterization. For instance, several existing algorithms are based on the observation that technical terms tend to be noun phrases (Frantzi et al., 2000; Justeson and Katz, 1995). However, again the converse is not true, and noun phrases are not generally technical terms. Consider for instance that *spare change* and *severe difficulty* have the exact same part of speech pattern that most of the technical terms described by Justeson and Katz (1995) and Frantzi et al. (2000).

We can generalize this to any of the characteristics that we observe of technical terms. Each of these observations regarding tendential characteristics of technical terms reveal clues that may be useful for recognizing technical terms, but we must not let ourselves be fooled into thinking that these clues are in any way exclusive to technical terms – these tendential characteristics are at best a matter of statistical correlation. Consequently, we would probably like to have access to as many such clues as we can when we attempt to extract technical terms. Note, for instance, that frequency of occurrence and part of speech pattern are unreliable indicators when used each on their own but are considerably more reliable when used together (Justeson and Katz, 1995; Manning and Schütze, 2000).

In particular, no one characteristic has proved to be sufficient as a indicator on its own (Vivaldi and Rodriguez, 2007).

### 2.3 Polysemy

Let us consider some arbitrary technical term and consider two distinction we can make about it. First, whether it has multiple reference in special language:

A) The term refers only to a single domain concept in special language.

B) The term refers to several domain concepts in special language.

Second, whether it is specific to special language:

a) The term does not occur in general language.

b) The term occurs in general language.

---

6 Although these phrases are, at least initially, specific to certain dialects or sociolects of English. It becomes considerably more difficult to find examples of neologisms that are not domain specific, since neologism almost invariably originate from specific domains, dialects, or sociolects.
Combining these two distinctions, we get four classes of terms:

Aa) A technical term occurring in only one special language domain that is not an ordinary word. Examples: support vector machine, electrocardiogram, γ-Aminobutyric acid

Ba) A technical term occurring in several special language domains that is not an ordinary word. Examples: entropy (physics and statistics), F-score (information retrieval, economics)

Ab) A technical term occurring in only one special language domain that is also an ordinary word. Examples: union, mechanics, capital, worm (computer security)

Bb) A technical term occurring in several special language domains that is also an ordinary word. Examples: dominance (behavioral biology, mathematics), static, complexity

Why should we care about this? After all there does not appear to be any conceptual difference between polysemy in this restricted sense and polysemy in the common linguistic sense. These different cases will, however, need to be treated differently if we try to find the point in time when a lexical unit came into use.

For the purpose of discovering technical terms using measures of neology, the difference in polysemy between A and B is not actually a general problem. After all, we only want to determine if the (surface form of a) term is a neologism, i.e. that there was some point in time before which the term was not in widespread use and after which the term started to appear rapidly. If the term is polysemous in special language but does not occur in general language then we might have difficulty determining when each sense started appearing, but we should not have any problems discovering that e.g. entropy has only been in use for the last century or so. By contrast, noticing that worm in the computer security sense is a relatively recent notion is considerably more difficult, since worm might also refers to e.g. earthworms, in which sense the word has been around for a very long time and is frequently used in common parlance. Since general language contribute an order of magnitude more to the total volume of human language, it is likely that the much more frequent general language senses will drown out special language senses completely.

Even though we should not have any problems finding out if a term of type Ba is a neology or not, we should still mention the opposing view due to Vivaldi and Rodríguez (2007): We must know the domain of the document to appropriately estimate the conditional probability that a polysemous term is used in (one of) its special language sense(s). Consider the word calculus. This word is a technical term in mathematics and medicine but does not have any general language senses. Consequently the probability that the lexical unit is a technical term is (close to) 1. However, by contrast consider the word agent. This is a term in linguistics, law, economics as well as in computer science. But it is also used in general language. Thus, in order to estimate the probability of the word being a technical term we must know the domain because this knowledge would change the conditional probability of discovering the term in the domain, and thus by Bayesian reasoning, the probability that it is a technical term.

2.4 Terminological Neology and Terminology Change

General language frequently forms new words and expressions through conversion or morphosyntactic processes, or import words and expressions via borrowing, but neologisms are
2. Background

seldom created de novo through pure invention in general language. Such word or term creation is generally done in special languages, after which the term might later be transferred to general language (Sager et al., 1980, p. 287).

Cabré Castellví et al. (2012) give some examples why new technical terms might be created:

1. A new entity is discovered or invented and thus lacks linguistic representation
2. When translating from a language where a given term exists into a language where it is absent
3. Language planning, when institutions have to decide what is the most appropriate term to use (i.e. prescriptive terminology)

They also point out the conceptual distinction between knowledge production on the one hand, and knowledge transfer between communities on the other (Cabré Castellví et al., 2012). Knowledge transfer in particular is generally the subject of deliberate language planning to a larger extent than knowledge production (Cabré Castellví et al., 2012). Neology in general language is mostly a spontaneous and undirected process, but in special language it is frequently a deliberate process. For an overview of the creation of new terms, we refer to Sager et al. (1980), which is based on withdrawn standards from ISO (ISO 704:1968, 1968) and British Standards (BS 3669:1963, 1963). For a more up-to-date but less thorough overview, see the ISO standard 10241 (ISO 10241:2011, 2011) which has since superseded the ISO standard 704.

In the general case, special language observes more rapid change than does general language, although the speed differs considerably between different subject fields. As an example of a quickly changing subject we have information science, where the rate of change has been observed to be as high as 4% per year, with roughly the same number of terms removed and added (Harris, 1979). As a consequence, any terminology resource for information science will become obsolete very quickly.

By contrast, the field of law is well-known to have a particularly low rate of change, and consequently a large part of its terminology has remained unaltered for centuries. Like the language in any discipline, and indeed like all special language, it is changing, albeit at a very slow pace compared to other disciplines (Lemmens, 2011).

The need for automatic extraction of terminology is consequently very different in quickly changing fields, since human terminologists will have a considerably harder time keeping up with the changes in terminology if a sizable part of the terminology has to be discarded and replaced with new terms every year. On the other hand, in a field where parts of the terminology is codified the need for automatic extraction is much smaller.

Processes of Term Creation

For an indepth treatment of the subject, we refer to Sager et al. (1980). We repeat the main points here, partly in order to maintain consistent terminology through the paper. In particular, note that Sager et al. define neologism as what we refer to as invention, whereas we consider neologism to be any newly formed phrase.

New terms may be formed by:

1. Morphological formation: Includes affixation: e.g. antimaoist, modernist, inflection: computational, or backformation: e.g. automate, automated from automatic
2. Syntactic formation: e.g. complex function, point of sale
2.4. Terminological Neology and Terminology Change

3. Compression: Removing parts of an existing term where the omitted parts can be inferred from common sense or context. E.g. *space allocation* from *allocation of space*, *steering wheel* from *wheel for steering*. If the relation between the determinant and the nucleus is obvious then the long form may never occur. The long form may also never occur if the relation between the determinant and the nucleus is complicated and therefore difficult to express, e.g. *latent Dirichlet allocation*.

4. Borrowing: Borrowing may occur either from general language, e.g. *computer worm* (computerscience), *lion* (heraldry), *saddle point* (mathematics) or from other languages, e.g. *langage/langue/parole* (linguistics), *eigenvalue* (mathematics), or from other special languages: e.g. *quark* (from James Joyce), *entropy* (from physics).

5. Acronymization: e.g. *SVM*, *Unesco*, *laser*

6. Conversion: The transfer of terms from one part of speech to another. E.g. *to pattern*, *token resistance*, *no say in the matter*. This is particularly prominent in English due to its relative lack of morphological markers of parts-of-speech. By contrast, in languages where verbs or adjectives require specific affixes any words that are converted to another part-of-speech will have to change its form.

7. Invention: Creating new terms without any antecedents, e.g. *hardware*, *telemetry*

In some cases terms may be formed through a combination of processes. In particular, note that in the case of compression, the noncompressed form often do not exist on their own but merely exist temporarily as a pattern for the compressed form (Sager et al., 1980).

When attempting to extract technical terms, we should consider whether to consider semantically equivalent, variational forms of a term to be the same term. If we use a frequency threshold of occurrence for terms, then term normalization may change our performance considerably. For instance, Daille (2005) reports that in her test approximately 10% of the hapax legomenon term candidates had variant forms within the candidate set, which might thus have been discarded from analysis with a frequency threshold set too low.

For syntactical variations we could use simple patterns for normalization, i.e. transforming all occurrences of common patterns such as *allocation of space* into *space allocation*. There is of course a risk that we introduce ambiguity or collisions, however, since this kind of compression also occurs for other patterns, such as *logic programming* for *programming based on logic* or *recursive descent parser* for *parser using recursive descent*.

The most common term normalization is probably in terms of simple morphological forms, i.e. the grammatical distinction between *computation*, *computations*, *computational*, *computationally*, et c.. Most authors do not mention whether they consider any of these as variational forms of each other, but the widespread reliance on POS tagging as a preprocessing filter seems to suggest that terms with different parts of speech are generally not considered as variants. On the other hand, most systems allow for simple plural/singular normalization (Kageura and Umino, 1996, p. 274). However, as always, we must consider that transferring a term to another part of speech sometimes changes its semantics. As an example, consider *procedure* vs *procedural*. Consequently, while semantically equivalent forms sometimes imply shared termhood (or lack thereof), this is by no means an ironclad rule.

---

7We leave it to the reader to decide whether the works of James Joyce constitute special language.
2. Background

2.5 Applications

We give some examples of applications where term extraction plays a central role:

1. Computer assisted translation (Aker et al., 2014; Dagan and Church, 1994; Vivaldi and Rodríguez, 2007)
2. Machine translation (Aker et al., 2014; Vivaldi and Rodríguez, 2007)
3. Information retrieval (Aker et al., 2014; Vivaldi and Rodríguez, 2007)
4. Ontology generation (Vivaldi and Rodríguez, 2007), Thesaurus generation
5. Knowledge management (Aker et al., 2014)
6. Semantic search (Aker et al., 2014)
7. LSP dictionary generation, bilingual terminology generation (Aker et al. 2014), glossary compilation (Vivaldi and Rodríguez, 2007), Term mapping (Pinnis et al., 2012)\(^8\)
8. Terminology generation (Nakagawa and Mori, 2003)

Applications of keyphrase extraction include:

1. Information retrieval (Medelyan and Witten, 2006; Nakagawa and Mori, 2003)
2. Document summarization (Litvak, 2008)
3. Book-of-book index construction (Nakagawa and Mori, 2003)
4. Document clustering
5. Document indexing
6. Document classification

It is generally remarked both for term extraction (e.g. Nazar and Cabré, 2012) and for keyphrase extraction (e.g. Turney, 1999; Weinberg, 1988), that the agreement between human annotators is low. It is then generally taken that this implies, by necessity, that any given algorithm can only achieve performance less than or equal to human annotators (Kim et al., 2010). However, for any given annotator we have a gold standard that is correct as far as that annotator is concerned. Rather than drawing the conclusion that the low inter-annotator agreement implies that the definition of terminology is vague, we can draw the conclusion that we are rather looking at a spectrum of problems rather than just a single one. After all, there is nothing that says that only a single definition can exist, and the apparent subjectivity might simply reflect a difference in needs. It might make sense to treat the whole problem as a supervised learning problem, where the task is to be able to reproduce the judgment of any given annotator, given a dataset labelled by that annotator. One downside to that approach, obviously, is that we would have to abandon unsupervised algorithms and have to rely on manually labelled datasets, which are expensive and time consuming to create.

\(^8\)Note the multitude of synonymous terms. This is an example where the terminology has not yet been standardized.
2.6 Term Extraction

If there is a difference in needs between different annotators, or similarly between domain experts and terminologists, then it might make sense to avoid performing intrinsic evaluation altogether, given that it is highly susceptible to the different judgments of the individual annotators. Extrinsic evaluation on tasks on which prior labelling of the technical terms is assumed to improve performance would eliminate the problem of annotator subjectivity. Provided the extrinsic task imply a consistent definition of technical terms, then high extrinsic performance would imply that the underlying technical term extractor has managed to distinguish between terms and non-terms under whatever definition of technical terms the extrinsic task is implicitly using.

2.6 Term Extraction

Unlike the case for keyphrase extraction, gold-standards are either not publicly available, or are not in widespread use (Aker et al., 2014; Chaimongkol and Aizawa, 2013; Vivaldi and Rodríguez, 2007). Consequently evaluation tends to differ from paper to paper on a more or less ad-hoc basis and evaluation generally depends on what resources the authors have available. Some general strategies used in evaluations are as follows:

Evaluation type i. Manually annotating existing datasets

This approach is expensive and time consuming, but has been done independently by several authors. We list those that we have found in existing literature:

The Chaimongkol dataset (Chaimongkol and Aizawa, 2013) is a fairly small dataset consisting of approximately 19,000 tokens extracted from the abstracts of the documents in the SemEval-2010 keyphrase extraction task dataset. The documents were exhaustively annotated manually by two human annotators of unspecified expertise, with a reported Cohen’s $\kappa = 0.48$. The dataset and the annotations are not publicly available, but have been gracefully provided for use in this project.

Vivaldi and Rodríguez (2007) used two corpora from the IULA Technical Corpus (Badia et al., 1998) containing approximately 100,000 and 10,000 tokens respectively. The two corpora were both exhaustively annotated by three specialist annotators, with a reported full inter-annotator agreement of 37% among all three annotators. The dataset is publicly available but not the annotations.

Pinnis et al. (2012) used corpora in Latvian and Lithuanian of unspecified size and a corpus in Croatian containing 15,603 tokens. Each corpus was manually annotated by an unspecified number of annotators with unspecified expertise. No inter-annotator agreement is reported. The datasets are not publicly available.

Aker et al. (2014) used a manually labelled gold standard. The dataset contained a single, large (over 2000 words) article in German, Hungarian, and Latvian, manually annotated by two translators and terminologists. They do not mention whether the documents were each annotated by more than one annotator, and no inter-annotator agreement is reported. The dataset is not publicly available.

Evaluation type ii. Ad-Hoc labelling

In this strategy, the authors design and implement an algorithm which they then apply on some dataset. The term extracted by the algorithm is then labelled by a terminologist, or more commonly, by a domain expert. The absence (and indeed impossibility) of gold standards here
2. Background

means that results are generally not comparable with each other (Hasan and Ng, 2014; Vivaldi and Rodriguez, 2007). Since gold data is thus only available for those terms output by the algorithm, results can only be given in terms of precision and not recall (Chaimongkol and Aizawa, 2013; Vivaldi and Rodriguez, 2007).

Evaluation type iii. Using precompiled terminologies

For some domains, handcrafted terminologies exist which can be used as gold standards. An example is the Unified Medical Language System (UMLS) for the medical domain (Bodenreider, 2004). Given a document in the same domain, we then assume that only those lexical units in the document which occur in the terminology are technical terms, and that all others are not. These assumptions give us an exhaustively labelled dataset, and consequently a way to evaluate in terms of both precision and recall. There are a few problems here however. First, the terminology is most likely not complete in the sense that all technical terms in the domain are contained in the terminology. It is reasonable to assume that any given terminology misses some terms, either because the data the terminology was generated from does not cover these terms or because the terms are newly coined. Second, the domain of the terminology and the domain of the dataset are assumed to coincide perfectly. This might not be true, for instance if the dataset contains documents from related interdisciplinary fields. Even for documents with the same domain as the terminology we should expect some bleedover from related fields (Loginova et al., 2012). If we do not assume these two assumptions to hold, then we would need to resign ourselves to evaluate simply in terms of recall. We also have the problem that it is hard to draw any conclusion from the results since the algorithm can only retrieve those terms that occur in the dataset. Thus, in order for the results to make sense as a measure of recall, we must ensure that the terms of the terminology all occur in the dataset (Loginova et al., 2012).

Existing systems

Early approaches, including Lexter (Bourigault, 1994), Terms (Justeson and Katz, 1995), acabit (Daille, 1996), Trucks (Maynard, 2000), the C-Value/NC-Value method (Frantzi et al., 2000) generally used POS tag based filtering combined with intradocument statistics. For an overview for most of these, we refer to Vivaldi and Rodriguez (2007). These are all evaluated using ad-hoc evaluation (type ii) (Vivaldi and Rodriguez, 2007, p. 233).

FASTR (Jacquemin, 2001) uses term normalization to spot term variants and is not a term extractor in the sense we generally use the term here.

Georgantopoulos and Piperidis (2000) use a non-deterministic finite automaton to find grammatical patterns which they then rank using statistical measures and evaluate using a preconstructed terminology (type iii).

Yate (Vivaldi and Rodriguez, 2007)(Vivaldi and Rodriguez, 2001; Vivaldi et al., 2001; Vivaldi and Rodriguez, 2007) is a hybrid system using ontology information, semantic context, morpheme decomposition, and mutual information. These disparate sources of information are then combined used voting and boosting. The system is evaluated on a manually annotated dataset (type i evaluation). The evaluation results are given as precision/recall curves and we consequently refer to the original paper for the results. The system is publicly available, but only works for Spanish and Catalan.

Most recent implementations tend to work fairly similarly to each other and generally use various kinds of document statistics combined with POS tagging.
2.7 Keyphrase extraction

Index terms, subject terms, and descriptors are a set of closely related concepts originating in library science which has since been adopted in information retrieval. These are the entries in the index section at the back of a book, or the entries in the subject indices used for library lookup. These two senses generally differ in focus, and a good set of candidates for one of these two purposes may not be a good set for the other. Still, the automatic extraction of either is approached using the same methods. Index terms in library and information sciences form what is known as a controlled vocabulary. These are carefully selected manually to merge synonymous terms in order to increase recall, and to disambiguate different word senses in order to increase precision.

Keywords, sometimes called keyphrases, are terms assigned to articles, conventionally by its author, that intend to summarize the main topic of its contents. While keywords and index terms are not generally used interchangeably, there is considerable conceptual overlap between the two terms. Keywords, much like index terms, are commonly used for searching collections of technical and scientific papers. Automatic keyphrase extraction is the task of automatically extracting a set of keyphrases from a document. Unlike human annotators, automatic keyphrase extractors are generally only expected to extract terms from the documents that actually occur in the text. Note that there is a difference in purpose between keyphrases, which generally consist of a handful of highly descriptive terms, and back-of-the-book index terms, the list of which commonly run several pages long and are often less topically descriptive than keyphrases. Despite this, index generation is often cited as a possible application of automatic keyphrase extraction (e.g. see Nakagawa and Mori, 2003).

In contrast to the dearth of public evaluation datasets for term extraction, there is no lack of datasets for automatic keyphrase extraction, which might partly be explained by the fact...
that there exist a large number of databases of technical and scientific papers with author or reader-assigned keyphrases. However, that a keyphrase was assigned by a human is by no means a guarantee of its appropriateness. For instance, human annotators have few reasons to assign only terms that actually appear in the text (Kim et al., 2010). Such keyphrases also generally only contain a single form of each term when multiple alternative lexical or syntactic forms exist.

Relevant datasets for technical and scientific texts include:

1. The ntcir tmrec corpus (Kageura et al., 1999)
2. The Inspec dataset (Hulth, 2003a)
3. The SemEval-2010 keyphrase extraction shared task (Kim et al., 2010)

Keyphrase extraction systems generally works in two steps: a) a preliminary filtering step where a list of potential candidates is identified in the document using various heuristics, b) a classification or ranking step where the preliminary candidates are narrowed down to a set of final output keyphrase candidates using either supervised or unsupervised machine learning or some kind of ranking algorithm.

The preliminary filtering is generally done using one of three approaches (Hasan and Ng, 2014):

1. Extracting all \( n \)-grams up to \( n = 3 \) that do not begin or end with a stopword.
2. Extracting all noun phrases.
3. Extracting all \( n \)-grams that match some predefined POS patterns.

Hulth (2003a) has done an evaluation of each of these approaches, and each of them achieve roughly the same performance, with no clear winner. She also demonstrated that using the majority vote of these three approaches yield somewhat better results than using any one in isolation. The choice of method in the preprocessing stage is often seen as a secondary concern, and the ranking stage has received a lot more scrutiny in the literature (Wang et al., 2014). Regardless of which method is ultimately being used however, authors commonly use some kind of additional filtering that removes stopwords, punctuation, mathematical formulae, and “semantically meaningless symbols” (ibid. p. 165). We should note that the filtering step of course has no understanding of the text and thus cannot determine what is meaningless and what is not. This filtering is generally done more or less unquestioningly (for a typical example, see Rose et al., 2010, p. 5) under the assumption that keyphrases generally do not contain any of these tokens. It is however easy to construct counterexamples. For instance consider terms like principle of least surprise, the real line, A*, tf.idf. Also note that the 'A' in 'A*' might end up being considering a stopword depending on the particular implementation.

We could make a distinction here between a) filtering that primarily removes difficult cases for the ranking step and b) filtering that primarily removes easy cases for the ranking step. If we are only removing the easy cases, those that the subsequent classification would have labelled correctly anyway, then the only thing we have gained is an decrease in runtime with a non-positive increase in evaluation performance. On the other hand, this filtering also has the potential to remove recurring noise that the classification might end up confusing for terms. Consider for instance that the initial document processing (e.g. the conversion from pdf to text) consistently outputs all page headers of a certain document as the same sequence
of garbage characters. Since these would then occur frequently in a single document, but never in any others, measures such as tf-idf would consider these highly relevant to the document.

Wang et al. (2014) has recently evaluated different variations of filtering strategies. They construct two improved datasets from the Hulth (2003a) and the SemEval-2010 datasets by removing articles where few of the keyphrases occur in the text, and by removing keyphrases not occurring in the text from the remaining documents. The resulting datasets thus have over 97% coverage of keyphrases. They then compare the coverage of the above three filtering methods. They report that each of these filtering techniques achieve around 65-70% coverage on the Hulth (2003a) dataset and around 70-85% coverage on the SemEval-2010 dataset. The filtering using n-grams achieve significantly lower coverage on all tests, but it is unclear from the paper whether this is due to the choice of method or because the authors used stemming for the other two methods, but not for the n-grams method. The main cause of the errors, however, was found to be that the filtering methods either only output candidates that covered part of the true keyphrase, or candidates that only contained the true keyphrase (i.e. the candidates were too long). Only 0.8% of the true keyphrases were found to contain punctuation, and only 1.8% were found to contain stopwords.

Given a set of candidates extracted it is then straightforward to apply standard supervised machine learning algorithms, provided labelled examples exist. The first such attempt (El-Beltagy and Rafea, 2009) was made by Turney (1999) using decision trees. A system that bears particular mention is KEA (Witten et al., 1999). KEA is an early keyphrase extraction system, and despite its relative simplicity is still quite competitive and is consequently often used as a baseline for evaluation (Hasan and Ng, 2014). KEA uses a primary filter which extracts all n-grams up to n = 3 that does not begin or end with a stopword. The system then uses naive Bayes to extract keyphrases using two features: tf-idf and the position of the first occurrence of the term in the document, normalized by the document length.

Subsequent supervised approaches to keyphrase extraction are almost invariably very similar to early systems such as KEA, and a multitude of different machine learning algorithms have been used by various implementations. Applying supervised machine learning algorithms to labelled examples is of course a completely straightforward exercise and does not concern us here. For an overview of various algorithms commonly used we refer to Kim et al. (2010) and Hasan and Ng (2014).

More interesting is the choice of features to use for supervised learning. The general trends are:

1. Intradocument statistical features: Primarily tf-idf (Hulth, 2003a; Turney, 1999, e.g.).
2. Position: The relative position of the first occurrence of the term in the document (KEA) is one common and performant feature. Several authors also use the structural position of the term, i.e. giving more weight to terms that appear in certain parts of the document, primarily the abstract in the case of technical and scientific papers (Hulth, 2003a; Turney, 1999).
3. Syntactical features: E.g., the most frequent POS tag assigned to the phrase in the dataset (Hulth, 2003a).
4. Features derived from external resources such as Wikipedia (Medelyan et al., 2009), or domain ontologies (Hulth et al., 2006; Medelyan and Witten, 2006).

Tf-idf is the single most commonly used feature, but it is not without its share of problems. Most importantly, for us to be able to use it as a feature we need to have a dataset that is large
2. Background

enough for the idf measure to be useful. The other common features listed above also force us to adopt certain assumptions about the data under consideration that may or may not be reasonable in the document's domain. For instance, document position make sense for news articles and scientific papers, but may not be as useful for e.g. meeting transcripts (Liu et al., 2009).

Unsupervised methods have also been used, and the state-of-the-art implementations of unsupervised methods have been demonstrated to achieve performance comparable to state-of-the-art supervised implementations. For instance, the third best system in the SemEval-2010 evaluation is KP-Miner (El-Beltagy and Rafea, 2009) which achieved an \( F_1 \) score of 25.2 compared to the best performing system, humb (Lopez and Romary, 2010), which achieved an \( F_1 \) score of 27.5.

We describe some of the later approaches that do not use standard machine learning techniques:

Tomokiyo and Hurst (2003) proposed an unsupervised approach using language models. They constructed an \( n \)-gram language model based on a general language corpus, which they called the background language model, and an \( n \)-gram language model based on the dataset they extracted keyphrases from, which they called the foreground language model. They then used Kullback-Leibler divergence between these different language models for finding candidate keyphrases. They used two distinct concepts they call phraseness and informativeness, which closely correspond to unithood and termhood respectively. They defined phraseness (unithood) for an \( n \)-grams as the KL-divergence of the probability of the \( n \)-gram under the foreground language model when constructed over \( n \)-gram against the same model constructed over unigrams. Intuitively, they thus define phraseness as how much more (or less) probable the \( n \)-gram is when considered in its entirety vs how probable each of the individual words that constitute the \( n \)-gram are. They defined informativeness (termhood) as the KL-divergence between the probability of the \( n \)-gram under the foreground language model against its probability under the background language model. Intuitively, they thus defined informativeness as how much more (or less) probable the \( n \)-gram is in the dataset than in general language. They evaluated on an unlabelled dataset and thus cannot give any indication of precision and recall. It is worth pointing out that this approach is quite a rare beast in that it does not use the standard methods of generating initial keyphrase candidates. This alternative preprocessing step does not appear to have been quantitatively compared to the common heuristics, apparently under the assumption that heuristic approaches are sufficient (Hasan and Ng, 2014).

KeyCluster is a clustering based approach proposed by Liu et al. (2009) that decompose documents into a set of word clusters. Each cluster is then assigned one representative keyphrase, and each document is assigned the set of keyphrases that represent the topics constituting the document.

A recently popular approach is the graph-based ranking. The first implementation was TextRank (Mihalcea and Tarau, 2004), but several variants has since been implemented. In graph-based ranking, each word is represented by a vertex and two vertices are connected by an edge if and only if the corresponding words occur within a predefined window size of each other in a document. A modification of the PageRank algorithm (Page et al., 1998) is then used to find words which are important in terms of the resulting PageRank score. Several minor

\[9\] Since they do not quote any terminology/terminography related papers, it seems likely that they were not aware that these exact same concepts had been defined in linguistic circles a decade prior.
modifications of TextRank has been proposed, including SingleRank and ExpandRank (Wan and Xiao, 2008). However, Hasan and Ng (2014) has subsequently tested TextRank, SingleRank, ExpandRank and KeyCluster on four datasets: DUC, Inspec, NUS, and ICSI, and none of the algorithms was found to be able to beat a tf-idf baseline. On DUC, TextRank and KeyCluster perform significantly worse than the baseline. On Inspec, TextRank performs significantly worse than the baseline. On NUS and ICSI, all algorithms perform significantly worse than the baseline. Subsequently, another graph-based approach has been proposed, TopicRank (Bougouin and Boudin, 2013), which was shown to outperform baselines consisting of tf-idf, TextRank and SingleRank on three datasets (SemEval, WikiNews, and DEFT), but fail to outperform tf-idf on Inspec.

Rake (Rose et al., 2010) is a simplification of TextRank which uses the degree of the individual vertices in the graph and the word frequencies instead of calculating the PageRank, which is a costly operation. The evaluation of this system was done identically to how TextRank was evaluated, i.e. only on Inspec with $T = 0.33$ and was found to achieve similar performance despite being much simpler to implement. Given the lackluster performance of TextRank, however, this result is somewhat underwhelming.

Dredze et al. (2008) use latent semantic analysis and latent dirichlet allocation to find the underlying topics that constitute the emails in the Enron dataset. They then use the intuition that good keyphrases generally are terms that characterize the latent topics of the email, and extract as keyphrases those terms which are most closely related to the email content in the LSA variant, and the terms with the highest posterior predictive probability in the LDA variant. Note that this approach can thus generate as keyphrases any term which appears in the email collection and the keyphrases do not actually have to be included in the email. This last point is, however, not mentioned by Dredze et al., and they do not consider keyphrase candidates outside each email's contents in their evaluation (ibid. p. 3).

Several authors use other documents in the collection to find either topics shared by groups of documents (Dredze et al., 2008) or clusters in the document collection (Wan and Xiao, 2008). The motivation for this is that is quite easy to find words and terms that only occur in the current document, but it is likely that these terms are chance occurrences such as numbers or misspellings. What we probably want is terms that occur seldom but still relatively frequently. More generally, keyphrases occur in varying degrees of descriptive granularity. For instance, At the most course-grained level we have general subject headings such as Mathematics or Biology. At more fine-grained levels we generally want keyphrases that distinguish between documents within clusters, rather than find descriptors that are common to each cluster. This is important if we want keyphrases that are useful for searching document collections – retrieving all documents in a collection that are labelled with a general subject heading is much too coarse grained for document retrieval (Boguraev and Kennedy, 1999; Weinberg, 1988). The degree of granularity in the description that we aim for will thus determine what constitute good keyphrases, and must thus also influence the choice of approach we adopt.

---

10Although the advantage over tf-idf in $F_1$ in each case is rather modest around 1.5 units of percentage.
11Intuitively: the term that would have been most likely to appear in the email, given the rest of the email collection and assuming the LDA generative model.
12Note that for the similarity measure between two terms to be defined for LSA, both terms have to occur in the data used to generate the LSA subspace. The case is similar for the posterior predictive in the LDA case: for all other terms the posterior predictive is undefined.
Google has been digitizing a large number of books over the past several years, and by the latest count the total number of books reaches 8,116,746 books, which allegedly constitutes over 6% of all books ever published (Lin et al. 2012). The selection of books is broad, and covers both general as well as special languages. For an in-depth overview of the Google Book project we refer to its official description, but we will repeat the parts most relevant to this study here.

The Google Ngrams dataset is a byproduct of this effort, and tracks the $n$-gram count occurring in the Google Books dataset for a subset of the languages for which books have been scanned. At the time of writing the languages for which $n$-gram counts are available are: English, Spanish, French, German, Russian, Italian, Chinese, and Hebrew, although the focus is primarily on the English version, which constitutes the majority of the data, both in terms of number of books (ca 56%) and number of tokens (ca 54%) (Lin et al., 2012).

Google Ngrams is not the only dataset of $n$-gram frequency counts, and $n$-gram statistics has long been a staple for building language models. However, the one thing that sets the Google Ngrams datasets apart from such similar $n$-gram datasets is that it not only provides frequency counts for each $n$-gram, but separates each data point into individual frequencies per year. Part of the reason for this is of course that the dataset used to construct the $n$-gram statistics contain metadata including the publication date for each book, and span over a very long period of time, which makes it possible to extract such time series. This time series data is useful for a wide variety of tasks where one wants to track trends or other kinds of temporal phenomena in word usage, such as examining cultural and social trends (see for instance Michel et al., 2011).

The data can be accessed using a web interface. The full data can also be downloaded, although the total size of data measures in several terabytes and is thus difficult to store and use. For an in-depth but popular overview of the dataset we refer to the description of the web interface. For a technical description we refer to the paper describing the extraction process of the most recent version (Lin et al., 2012).

The data include all $n$-grams up to $n = 5$, although $n$-grams occurring less than 50 times are excluded due to space constraints. In the most recent version (2012), the $n$-grams do not span sentence boundaries but do span page boundaries. Sentence boundaries are marked by the special symbols _START_ and _END_. Tokenization has been carried out in a fairly straightforward manner: punctuation is considered individual tokens, contractions like don’t is expanded into do not, and hyphenated compounds like case-sensitive is split into case-sensitive, i.e. three tokens with the hyphen considered a separate token. The non-latin scripts, i.e. Cyrillic and Chinese, but also other languages that occasionally appear in the data, are retained as unicode in the data but are transliterated into ascii for the sake of collation.

The preprocessing also include POS tagging and dependency parsing. Most tokens in the data thus include its POS tag label (in Penn Treebank format), but for a large number of entries this information is missing. The dependency information is not used in the main dataset, but is contained in a separate part of the data, called the "0-grams". These are really a kind of bigram skipgrams, consisting of bigrams on the form $A \Rightarrow B$, where $A$ occurs in the subtree below $B$ in the dependency parsing. The data entries for these "0-grams" are otherwise identical to the regular $n$-gram data, including POS tagging and frequency counts.

http://www.google.com/intl/en/googlebooks/about/index.html
https://books.google.com/ngrams
https://books.google.com/ngrams/info
A related but separate dataset (Goldberg and Orwant, 2013) extends the 0-gram data to relations involving up to 4 participating tokens. This dataset, however, is smaller and does not currently have a web interface.

The dataset contains data from 1500–2008. The default setting of the web interface, however, is to only include frequencies inbetween the years 1800–2000. This choice mirrors the recommendation in the data description only to use this range. The reason to exclude the last eight years is that books published after 2000 are judged by different criteria when considered for inclusion into the Google Books project, and any analysis of the trends spanning the year 2000 must take this into account. The years 1500–1800 on the other hand suffer from data sparsity, since the number of books published during that period is much lower than in 1800–2000, and also from OCR-related problems due to different typographic conventions, such as the historical use of the long medial s, and from problems due to the extraction having difficulty parsing historical spellings.
In this section we will describe the work done in this study, both the technical implementation details, such as how we store and retrieve the $n$-gram data and how we handle the gold standard datasets, as well as the more scientifically pertinent questions concerning how we will extract meaningful information from the $n$-gram data that we have.

This section mostly serves to document the implementation details that underlie Sections 4 and 5. These are generally written to be self-contained, and the reader can well skim through this section and subsequently refer to it if the finer details of implementation become necessary.

3.1 Indexing Backend

In order to analyze the neology of technical terms we need some data to base our analysis on. In this work we will use the Google Ngrams dataset but there is nothing stopping us from using other temporally labelled datasets. The Google Ngrams dataset is derived from the Google Books dataset and thus focuses on general language, but it does have the advantage of having the frequency counts already extracted.

The original data consists of term frequencies of one datapoint per year and $n$-gram, from 1500 until 2008, as well as document frequencies (the number of individual books in which the $n$-gram appeared) in the same time range. The $n$-grams are also separated based on its pos tag. Thus, for each $n$-gram with a specific pos tag we get at maximum of 509 x 2 scalar entries, although the data is sparse for many of the $n$-grams.

What amount of the original data we need to store, however, depends on what we intend to do with the dataset. If we have already decided on what features to use, then we can simply extract and store those features, which brings the data size down to about one thousandth of its original size in the optimal case. Conceptually this is similar to the probably most common way to use the dataset, by merging all the datapoints for a single $n$-gram into one aggregate number. If on the other hand we want to experiment with the dataset in order to extract and compare different features, as we do in this study, then we have to resort to storing as large a part of the dataset as is required for the features we want to examine. In this study we restrict ourselves to the time range 1800–2008, which is probably the one of most concern to us as we investigate the neological properties of technical terms. We also restrict ourselves to the term frequencies and discard the document frequencies. This brings in the optimal case the data size down to about one fourth of its original size, although in reality the actual reduction is much less, since many of the entries do not contain data for all years.

We use Lucene for storing the data, which allows us to compress the entries in order to save disk space, but still allows us prompt retrieval of individual entries. We store each $n$-gram as a document so that we can retrieve them individually. This necessitates us to avoid using Solr but rather interface directly with the Lucene API to avoid the overhead of Solr’s REST-like API and document handling. This overhead would have been severe when we have myriads of single word documents, each of which would have had to be wrapped in XML or JSON and transferred via HTTP. Skipping Solr means that storing a single $n$-gram entry is just a single java function call.

To check how much space would be required to store the reduced data, we simply index this data and check the resulting file size. In the trigram case we index the data for the trigrams
3. Method

|          | R1      | R2      |
|----------|---------|---------|
| unigrams | 5.7 GB  | 4.0 GB  |
| bigrams  | 140.2 GB| 84.7 GB |
| trigrams | 2,543.2 GB† | 820.7 GB† |
| 4-grams  | 919.5 GB† | 130.7 GB |
| 5-grams  | 840.0 GB† | 94.7 GB  |

Table 3.1: Datasize for storing the compressed data, reducing the data by only considering term frequencies in the time range 1800–2008 (R1), and reducing the data by extracting the final features (R2). Entries marked with an obelus are estimated values.

beginning with the letters A–L. This part of the data requires roughly 391 GB to store. Assuming that the words in English beginning with the letters A–L constitute 47.631% of all words in the language, we can infer the presumed total size.

To check how much space would be necessary to store the full timelines in the period 1800–2008 we index the unigrams and bigrams and estimate the trigrams, 4-grams, and 5-grams by extrapolation. The TH data file for bigrams is 4.73 GB uncompressed, for trigrams 85.8 GB, for 4-grams 31.02 GB, and for 5-grams 28.34 GB. We assume that these proportions are representative of the whole dataset. Based on these numbers and assumptions we can thus extrapolate the remainder of the table (Table 3.1).

As we find out (Table 3.1), it is still impractical to store the trigrams even when reducing the dataset in this way. Furthermore, due to data sparsity in the Google Ngrams dataset for higher order n-grams only about a fourth of the trigrams that constitute technical terms occur in the dataset. Thus, it seems unlikely that technical term extraction would benefit from using information about trigrams and higher, and so we omit these from consideration in our system.

3.2 Problem Definition

It might be instructional to start by using an example. Let us look at one of the documents in the Chaimongkol dataset, with the annotated technical term spans highlighted (Figure 3.1).

This excerpt was of course labelled manually, since it is taken from the gold standard, and our task is to perform the same labelling without any human assistance.

As we elaborated in Section 2, early technical term extraction attempts worked fairly similarly to keyphrase extraction systems, using an initial n-gram or POS tag-based filtering to identify term candidates, then proceeding to narrow this list down using machine learning algorithms on various kinds of document statistics such as term frequency or the dice coefficient Frantzi et al. (2000); Justeson and Katz (1995); Pinnis et al. (2012). The main difference from keyphrase extraction is mostly a choice of top-level machine learning approach: in technical term extraction it makes sense to view the problem as a binary classification problem (Chaimongkol and Aizawa, 2013), whereas in keyphrase extraction it makes more sense to see the problem as a ranking problem.

---

1https://en.wikipedia.org/wiki/Letter_frequency

2We cannot extrapolate from the unigrams because the unigrams are split into separate files based on the initial letter of the unigram, whereas bigrams and above are split into files based on the initial two letters.

3For a list of examples, see Section 2.6.
Scalable Grid Service Discovery Based on UDDI

Efficient discovery of grid services is essential for the success of grid computing. The standardization of grids based on web services has resulted in the need for scalable web service discovery mechanisms to be deployed in grids. Even though UDDI has been the de facto industry standard for web services discovery, imposed requirements of tight replication among registries and lack of autonomous control has severely hindered its widespread deployment and usage. With the advent of grid computing the scalability issue of UDDI will become a roadblock that will prevent its deployment in grids.

Figure 3.1: An excerpt from one of the documents in the Chaimongkol dataset. Gold labelled technical terms are rendered in black, and the remainder of the text is rendered in grey.

If we were to assume the ranking approach the problem would reduce to the following: given a document, retrieve all technical terms that occur in it, in decreasing order of confidence. This approach is appropriate if, for instance, we are building a system for lexicon generation or document clustering, but less so if we want to retain the context of each term. There are other downsides, such as that the approach is ill equipped to handle nested terms, a routine occurrence for technical terms. We would also endow the system with the rather non-intuitive behaviour that the termhood or lack thereof for some arbitrary term would depend on the termhood of the term in the remainder of the document.4

While the ranking approach is common, we will here follow the example of Chaimongkol and Aizawa (2013) and construe the task as a sequence labelling problem. Thus, we are given a sequence of symbols and seek to label the symbols such that the relevant spans are identified.

In the case of two tag classes, the classes inside and outside spans are, in NER as well as in other similar sequence labelling tasks, conventionally denoted i and o, respectively. It is also common to label the first symbol in each span using a different token, in order to be able to denote spans that are adjacent to each other. These are conventionally labelled b, which give the tagging scheme its common name: BIO tagging (Sang and Veenstra, 1999).

However, when dealing with languages such as English, where the head comes at the end of the phrase, it makes more sense to use a tagging scheme where the span boundaries are marked by the last symbol rather than the first. After all, by giving it a different label, we will implicitly treat it differently. Thus we also consider a LI0 tagging scheme, where the last symbol is tagged with an l tag, and compare the performance to the performance for BIO tagging. Other more complex tagging schemes exist, such as BILOU, denoting beginning, inside, last, unit, and outside tags respectively (Ratinov and Roth, 2009), but we will not consider these here. For the vast majority of cases, the simpler IO tagging is most likely perfectly adequate.

In the sequence labelling interpretation of the problem, we see each token as a node in a

4Note that we most likely do want the termhood of an arbitrary term to depend on the other terms in the document – these are of course indicative of the document’s topic. The problem is that the ranking presupposes some cutoff, in other words, a document will always contain at most x technical terms, so an arbitrary term could be labelled a technical term in one document but not in another document of the same domain and on the topic, simply because the former uses other terms differently, even if the term itself is used identically in both documents.

5Unlike, for instance, French.
3. Method

linear graph, and the problem is to select one out of a predefined set of possible labels for each node, such that it maximizes the fit to the observed data, using some appropriate measure of fitness. There exist a multitude of algorithms for sequential labelling under different assumed models, beginning with the bread-and-butter Viterbi algorithm for hidden Markov models. We also have conditional random field models, which can be inferred using several different optimization algorithms such as for example BFGS. For a refresher on conditional random field models, hidden Markov models and their relevant training algorithms we refer to any standard textbook on statistical machine learning, for instance Murphy (2012). Hidden Markov models were popular in the nineties, but has since fallen out of favor. One of the downsides of hidden Markov models in sequential labelling is that a hidden Markov model makes a number of simplifying assumptions to make inference tractable, among them the assumption that the features (emissions) of each node are independent given the node label. For simple feature sets, and in toy problems, this is generally unproblematic, but in real world applications, where we often create elaborate feature sets to aid classification performance, we are very unlikely to have features that are anywhere near independent. More importantly, we would be unable to use variations on the same features, such as affix patterns of different length, or binned data using different bin sizes, without violating the conditional independence assumption. Conditional random fields on the other hand do not assume that the features are mutually independent and thus tend to be more robust in the face of richer feature sets (Sutton and McCallum, 2007).

More recently, structured averaged perceptrons, structured SVM and other methods have also been used with good results. We will not treat these here, but refer the interested reader to Nguyen and Yunsong (2007) for details.

Similarly to (Chaimongkol and Aizawa, 2013), we implement a CRF model using the freely available state-of-the-art CRF framework CRFSuite.6

3.3 Neological Features

We have noted that the shape of the timelines seem to indicate whether a given term is recent coinage, but in order to use these as input to machine learning algorithms, we need to distill the high dimensional data into low-dimensional features that retain the neological information.

![Figure 3.2: Unsmoothed time series for the bigram term extraction.](http://www.chokkan.org/software/crfsuite/)

What we want is something that is able to capture the kind of shape that begins with very low values and at some point starts to spike rapidly. We could for instance try first order derivatives and look for large increases. However, there is one major flaw in this approach. First order derivatives are, on their own, very sensitive to noise, and while it might work for terms where the noise is relatively benign, many technical terms are very infrequently used,

---

6http://www.chokkan.org/software/crfsuite/
and as a consequence have a very large amount of noise (Figure 3.2). We could of course try smoothing the data, but we are likely to retain the spurious spikes in the 19th century that we observe in Figure 3.2. The amount of books published has risen exponentially in the last few centuries, and in the 19th century published books were still scarce by modern standards. Thus, each book published in the 19th century constitutes a much larger fraction of the books published per year. In this case (Figure 3.2), just a handful of occurrences make it seem like the term came into vogue and disappeared twice in the 19th century. While ordinary smoothing or filtering might help in some cases, it is of much less help when the signal-to-noise ratio is low (Figure 3.3). In this case, it does amplify the signal over the noise, but first order derivative based features might still pick up the spurious spikes in the 19th century. One can do better by noticing that the spikes generally occur in single years and as a consequence, that the minimum of \( n \) consecutive years should be more representative of the underlying signal (Figure 3.4) than the average, and in particular, less likely to be influenced by single year spikes. This might work less well if the time series contain inverse spikes (dips), but this does not appear to be a concern in practice.

Alternatively, we could try to look for the maximum value of the curve, but again, this runs into the same problem. For instance, in Figure 3.2, the highest point on the curve is the one for \( t = 1808 \), corresponding to just two books.

What we want to extract is, however, not necessarily the shape of the timelines, but whether the occurrences of the \( n \)-grams predominantly occur in the far right side on the time axis. In

---

\( ^7 \)The spikes correspond to only two distinct mentions: First, To the solution of any of these substances, while the others remain undissolved, the term extraction is applied; and when, by evaporation, the substance extracted is reduced to a solid form, it is termed an Extract, which is hard or soft, watery [...], from The Edinburgh new Dispensary published in 1803, republished twice and quoted 12 times in other publications. Second, Surgeons also sometimes apply the term, extraction, to the removal of tumours out of cavities, as for instance, to the taking of cartilaginous tumours out of the joints; [...], from the Dictionary of Practical Surgery published in 1809, and republished twice. Of course, term extraction is not a unit in either of these cases.
other words, we want to determine if the timeline is mainly concentrated on the right side. This is simple to do using statistical measures such as the mean and the standard deviation of the curves.

Let $f_y^i$ denote the frequency of an $n$-gram $i$ in year $y$. Then

$$p_i(y) = \frac{f_y^i}{\sum_y f_y^i}$$

constitutes a probability density function, with the expected value:

$$\mu_i = \sum_y p_i(y) \cdot y = \frac{\sum_y f_y^i \cdot y}{\sum_y f_y^i}$$

How this "mean" should be interpreted might not be completely intuitively obvious, but for our purposes here it is enough to note that $\mu_i$ indicates where the curve is mainly concentrated. If the curve is concentrated around higher values of $y$ then we have, by definition, a surface level neologism.

We can take the standard deviation $\sigma_i$ of $p_i$ in the same way:

$$\sigma_i^2 = \sum_y p_i(y) \cdot (y - \mu_i)^2 = \frac{\sum_y f_y^i \cdot (y - \mu_i)^2}{\sum_y f_y^i}$$

The standard deviation $\sigma_i$ then yields a measure of how much the probability density is concentrated around $\mu_i$, in other words, how "steep" the probability density is. A low standard deviation consequently indicates that the term has been adopted or abandoned rapidly. Low standard deviation should thus in general imply either surface level neologisms or fads. If we are only interested in how quickly a term has been adopted and not how quickly it might have been abandoned, then we can take a one-sided "standard deviation", by separating out the $y$ for which $y < \mu_i$, but this does not seem to make much difference for the sake of the separability of technical terms. If a term falls into relative disuse, it tends to do so relatively quickly. Note, however, that terms seldom disappear entirely. As an example, consider the term *phlogiston*. While the physical sciences has abandoned phlogiston theory entirely, the term still turns up in contemporary literature, if only to discuss that the term was once used.

The skewness of $p_i(x)$ follows similarly:

$$\gamma_i = \sum_y p_i(y) \cdot \left( \frac{y - \mu_i}{\sigma_i} \right)^3 = \frac{\sum_y f_y^i \cdot \left( \frac{y - \mu_i}{\sigma_i} \right)^3}{\sum_y f_y^i}$$

Skewness is a statistical measure of the inclination of the distribution to be concentrated to the left or to the right of the mode, or more simply put, whether it slopes to the left or to the right, and should thus in theory capture precisely the characteristic of the curves that we are interested in. Unfortunately however, and as we will see later on, the skewness is not a very useful indicator of technical terms, possibly because of problems due to noise.

As an illustration of the correspondence between neology and the mean and standard deviation, we refer to Figure 3.5. Time series for $n$-gram that are declining in usage or stable over time tend to have means around the center/left of the x-axis, and tend to have large standard
deviation.
deviations. By contrast, neologisms tend to have means towards the far right of the x-axis, and tend to have much lower standard deviations. In general, terms that are falling out of favor do so much more slowly than ascendant terms are being adopted. Thus, while we should expect lower time series means and standard deviations for these terms than stable terms, the difference is not nearly as marked as the difference between neologisms and stable terms.

![Figure 3.5: Time series for the unigrams upon (top), and (middle), and algorithmic (bottom), with the relevant mean and standard deviations marked.](image)

We should hasten to point out that there does not seem to exist any theoretical reasons to use these statistical measures in this way. For instance, using the peak of the curve (i.e. the mode of the distribution) might have more intuitive appeal, since this should correspond to the point in time at which the term was in its most widespread use. However, the Google Ngrams dataset is often plagued by severe noise, in particular for less commonly used n-grams such as technical terms, and the peaks of the timelines are thus likely to be spurious. The mean and the standard deviation may be crude measures of shape, but they have the advantage of being robust against noise, and can generally be used with good results even for very noisy timelines.

Beyond these, other intuitively appealing features have been tried, such as features based on first order derivatives of the timelines, but these have not been useful, again presumably due to the noise.

### 3.4 Baselines

We will compare the performance of the neology features to ten different baselines:

1. **Raw words**
2. **Lemmas**
3. Method

3. Lowercased words

4. Word shape features extracted similarly to Chaimongkol and Aizawa. These include binary features such as whether the current token is capitalized, uppercased, or alphanumeric. For a complete list with examples, see Fig 3.2.

5. Affixes of length up to 4 characters extracted for all tokens. In other words, for the token carbonization we would extract carb-, car-, ca-, -tion, -ion, -on, and -n.

6. POS tags using the Stanford POS tagger. 9

7. Rank in a pregenerated list of words ordered in descending order by frequency of usage. Thus the nth most common word in the English language would have rank equal to n. Filtering out words having rank less than some predefined threshold is thus equivalent to using a stop list.

8. Mutual information between each consecutive bigram.

9. Generalized dice coefficient between each consecutive bigram.

10. TF-IDF for each unigram and bigram in the dataset.

In previous studies on related problems, affixes are usually (Stenetorp et al., 2011) included as one of the word shape features, but we will separate it out here, since it has very different performance compared to other word shape features.

We can generally divide the above features into two classes. Words, lemmas, lowercased words, wordshapes, affixes, and POS tags on the one hand require us to learn which feature values are indicative of technical terms and which are not. In other words, these can only be used in supervised machine learning. 10 On the other hand, rank, MI, GCD, TF-IDF, and neology are features where higher values can be assumed to be indicative of technical terms. Consequently, these can also be used in unsupervised machine learning.

Because CRFSuite cannot handle continuous features, such as tf-idf, μ, σ, or γ, we have had to resort to discretizing these by binning. Appropriate bin sizes have been established experimentally.

---

9http://nlp.stanford.edu/software/tagger.shtml

10Although the POS tags indicative of technical terms are quite straightforward and do not require much training data to infer.
| Feature                  | Sample input | Sample output |
|--------------------------|--------------|---------------|
| Pattern                  | SemEval-2010 | AaaAaaa-0000  |
| Camel case               | camelCase    | aaaaaAaaa     |
| Collapsed pattern        | SemEval-2010 | AaAa-0        |
| Camel case               | camelCase    | aAa           |
| Only uppercase characters? | CRF          | true          |
| Camel case               | camelCase    | false         |
| Only lowercase characters? | dielectric  | true          |
| Camel case               | camelCase    | false         |
| Only alphabetic characters? | 2nd          | false         |
| Only alphanumeric characters? | 2nd          | false         |
| Only digits?             | 2015         | true          |
| 2015-01-01               | false        |
| Contains uppercase characters? | SemEval-2010 | true         |
| Contains lowercase characters? | SemEval-2010 | true         |
| Contains alphabetic characters? | SemEval-2010 | true         |
| Contains alphanumeric characters? | SemEval-2010 | true         |
| Contains digits?         | SemEval-2010 | true          |
| Titlecased?              | SemEval-2010 | true          |
| Camel case               | camelCase    | false         |
| 2nd                      | false        |

Table 3.2: The individual features that constitute the word shape features, as well as some examples of input-output pairs.
In this section we will look at timelines of frequency over time for neologisms and non-neologisms and examine ways to extract features that capture the difference.

In this section, we examine statistically the features we propose, and show that both technical terms and keyphrases are strongly biased towards certain values for our features. We also underpin the relationship between technical terms and keyphrases by showing that these are very similar in terms of neology. We finally underscore that named entities can be seen as a subset of technical terms by showing that these too have similar neological properties.

4.1 Choice of \( n \)-gram order

We omit trigrams and higher order \( n \)-grams from consideration. It is unlikely that higher order \( n \)-grams would help in keyphrase and technical term extraction, because the Google Ngrams dataset excludes any \( n \)-gram that has a total frequency of occurrence less than 50. This means that less frequently used \( n \)-grams, such as technical terms, as well as higher order \( n \)-grams are likely to be missing (see Table 4.1). This problem is not very severe for unigrams and bigrams, but technical term trigrams suffer from data sparsity problems severe enough to essentially render them useless. Part of this problem might be due to the large mismatch between the dataset used for evaluation, and the Google Ngram dataset used to identify neology. If we were to use an external dataset from a domain more similar to the evaluation set, then we might expect to find a greater portion of the \( n \)-grams in the external dataset. This unfortunately falls outside the scope of this study.

|              | Technical terms | Background |
|--------------|----------------|------------|
| Unigram      | 97.5 %         | 100 %      |
| Bigram       | 85.0 %         | 97.0 %     |
| Trigram      | 25.5 %         | 71.0 %     |

Table 4.1: The ratio of \( n \)-gram in each class in the Chaimongkol-Aizawa dataset that occur in the Google Ngrams dataset. The percentages have been generated by choosing a random sample of 200 unigrams of each class, 200 bigrams of each et c., and checking if the \( n \)-gram occur in the Google Ngrams dataset using the web interface.

4.2 Class separation

To analyze keyphrases we will use the SemEval-2010 dataset (Kim et al., 2010), one of the most commonly used gold standards for keyphrase extraction. To analyze technical terms we will use the Chaimongkol-Aizawa dataset (Chaimongkol and Aizawa, 2013). This dataset consists of the abstracts from the SemEval-2010 dataset manually annotated by two annotators such that all the technical term spans in the dataset have been labeled. Since the Chaimongkol-Aizawa dataset was constructed from the abstracts of the SemEval-2010 dataset we assume that these datasets are similar enough that analyzing and comparing the statistical properties of these two datasets is meaningful.

For the analysis in the following section, we construct three classes of data:
4. Analysis

- We extract the $n$-grams that are part of the spans labeled as technical terms in the Chaimongkol-Aizawa dataset to obtain one set of positive examples of technical terms.

- We extract the constituent $n$-grams from the gold-standard keyphrases from the SemEval-2010 training dataset (using the combined set) to obtain one set of positive examples of keyphrases.

- We extract the $n$-grams that are not part of the spans labeled as technical terms in the Chaimongkol-Aizawa dataset to obtain one set of negative examples of both technical terms and keyphrases.

We could of course extract negative examples of keyphrases from the SemEval-2010 dataset, but it is not so clear that these would all unquestionably be negative examples of keyphrases. Given that inter-annotator agreement is generally very low for keyphrases, this set might well contain terms that could reasonably be considered keyphrases by other annotators. We here assume that the negative examples of technical terms also constitute negative examples of keyphrases, and that this set is less likely to contain borderline cases of keyphrases. Only using three classes of data also simplifies both the exposition and the processing.

For these three classes of $n$-grams, we extract the corresponding timelines from the Google Ngrams dataset over the period 1800–2008, and we calculate the means $\mu$ and standard deviations $\sigma$ from these as defined in the preceding section. To analyze the features $\mu$ and $\sigma$ we plot the histograms of the values for the technical term $n$-grams and the keyphrase $n$-grams versus the background $n$-grams (Figure 4.1).

In order for the features to be useful for either technical term extraction or keyphrase extraction we would like to see as little overlap as possible between the histograms of the positive and the negative examples. This seems to hold true for the bigrams, but not for the unigrams. In the unigram case we can see only a weak tendential difference in the histogram densities of the positive and negative examples.

We should point out that the mode of the histogram densities for the negative examples fall very close to the mean and standard deviation of a uniform distribution over the period 1800–2008. These would occur around 1904 and 60.48 respectively.

In all cases, the histograms for the technical terms and the keyphrases are very similar. This might not seem very surprising given that all the data is derived from the SemEval-2010 dataset, but it bears mentioning that these were annotated by different people, and more importantly, using very different annotation criteria.

4.3 Relevance and usefulness of individual features

We have thus far argued that the mean and standard deviation are highly indicative of technical terms and keyphrases, but it is still not clear that these are useful for technical term and keyphrase extraction in the presence of other features. It might be the case that the means and standard deviation do not add any additional information beyond what can be inferred from the other features. We should also be mindful of whether the mean and standard deviation are orthogonal or are highly correlated with each other. It might for instance be the case that the useful information is contained in a subspace of the combined feature space, and that we could use e.g. PCA to extract better feature representations. To investigate these questions we will now look at scatterplots of each pair of individual features.

If two individual features are mutually beneficial then we would expect that the separability of the technical terms and the non-technical terms would increase if we use the two features
4.3. Relevance and usefulness of individual features

Figure 4.1: Class separation between technical terms, keyphrases, and background terms over $\mu_i$ (left) and $\sigma_i$ (right), when considering unigrams (top), and bigrams (bottom). Histogram bin size was set to 2 in all cases. The resulting histograms have been smoothed using Matlab’s default settings (moving average with span 5) and normalized to sum to one.

together. In other words, given two continuous features $f_x \in X$ and $f_y \in Y$, a classifier trained on $(f_x, f_y) \in X \times Y$ should be able to achieve higher performance than a classifier trained on either $f_X$ or $f_Y$. To test this we will train a linear support vector machine as well as a logistic regressor on the feature pairs $(f_i^x, f_j^y)$. The choice of these two methods is due to the result that a support vector machine is guaranteed to converge to the (unique) placement of the decision boundary resulting in optimal binary classification accuracy given the constraint on the shape of the decision boundary imposed by the kernel function, and the logistic regressor is guaranteed optimal fit of the observed class probability (posterior predictive).

The support vector machine case is most likely the most straightforward, so why do we need another way to measure the separability? The problem is that it only gives a binary answer. It might well be the case that there exists some line on the one side of which data points are likely to be of one class, with a probability proportional to the distance to the line. Logistic regression is designed to model this state of affairs, and so it makes sense to use a logistic regressor here. There are methods for using the distance to the classification boundary in a support vector machine in a similar fashion, but support vector machines are designed to give optimal binary classification accuracy and give no guarantees about the quality of inferred probability. A logistic regressor on the other hand guarantees optimal regression fit, under the assumptions of the model, namely that the probability of observing each label follows a Bernoulli distribution, i.e. $p(x = 1) \propto \theta^{(x=1)}(1-\theta)^{(x=0)}$, with $\theta$ following a multivariate logistic function.

The separability in the support vector machine case is quite readily given by the support vector machine classification accuracy. The separability in the logistic regression case will here
4. Analysis

| Averages  | μ    | σ    | γ    | tf · idf | Rank | GDC  |
|-----------|------|------|------|----------|------|------|
| μ         | 64.7%| 75.4%| 71.2%| 69.3%    | 70.2%| 68.6%|
| σ         | 70.0%| 72.4%| 73.7%| 72.2%    | 68.0%|      |
| γ         | 59.6%| 66.2%| 61.2%| 59.0%    |      |      |
| tf · idf  | 60.8%| 68.2%| 60.8%|          |      |      |
| Rank      |      |      |      | 65.1%    | 63.5%|      |
| GDC       |      |      |      |          | 51.2%|      |

| Standard deviations  | μ    | σ    | γ    | tf · idf | Rank | GDC  |
|----------------------|------|------|------|----------|------|------|
| μ                    | ±3.1%| ±1.0%| ±1.2%| ±2.9%    | ±4.9%| ±1.2%|
| σ                    | ±1.6%| ±1.2%| ±1.6%| ±1.9%    | ±5.9%|      |
| γ                    | ±1.2%| ±1.4%| ±1.2%| ±1.0%    |      |      |
| tf · idf             | ±0.8%| ±1.3%| ±0.8%| ±0.7%    |      |      |
| Rank                 |      | ±0.8%| ±4.3%|          |      |      |
| GDC                  |      | ±0.3%|      |          |      |      |

Table 4.2: Classification accuracy for support vector machines trained on the same data as in Figure 4.2, non-parametrically bootstrapped using 10,000 samples and averaged over 10 trials. Higher scores are better.

| Averages  | μ    | σ    | γ    | tf · idf | Rank | GDC  |
|-----------|------|------|------|----------|------|------|
| μ         | 39.8%| 32.5%| 37.5%| 35.7%    | 37.8%| 39.9%|
| σ         | 38.8%| 37.7%| 34.2%| 36.4%    | 38.3%|      |
| γ         | 47.7%| 41.1%| 43.7%| 47.1%    |      |      |
| tf · idf  | 42.9%| 40.3%| 43.1%|          |      |      |
| Rank      |      | 45.8%| 45.4%|          |      |      |
| GDC       |      | 49.5%|      |          |      |      |

| Standard deviations  | μ    | σ    | γ    | tf · idf | Rank | GDC  |
|----------------------|------|------|------|----------|------|------|
| μ                    | ±0.8%| ±1.1%| ±1.1%| ±1.3%    | ±1.3%| ±0.8%|
| σ                    | ±1.2%| ±0.8%| ±0.8%| ±1.2%    | ±0.7%| ±0.7%|
| γ                    | ±1.1%| ±0.7%| ±0.7%|          |      |      |
| tf · idf             | ±1.3%| ±0.8%| ±0.8%|          |      |      |
| Rank                 |      | ±0.5%| ±1.0%|          |      |      |
| GDC                  |      | ±0.3%|      |          |      |      |

Table 4.3: Macro averaged regression squared residuals for the logistic regression in Figure 4.2, averaged over 10 trials (top) with their respective standard deviations for each average (bottom). All values are in percentage units. Lower scores are better.

be given as the square deviation, which is proportional to the log posterior predictive, that is to say, the probability of the data under the inferred model.

We should also point out that the features $\mu$, $\sigma$, and $\gamma$ seem to be closely correlated, and the meaningful information appears to lie in single-dimensional subspaces of the space spanned by the features. This should not be very surprising, given that for instance high means implies that the timelines are densely concentrated on the right side, which must of course also imply
4.3. Relevance and usefulness of individual features

a small standard deviation.

In each case we draw a non-parametrically bootstrapped sample from the dataset in order to estimate the distribution of observed values. We repeat the procedure 10 times in each case and report the observed mean and standard deviation.

We could use other kernel functions in the support vector machine to model other shapes than linear decision boundaries, but this would not add much to the analysis. Our purpose here is of course to give an indication of how the features interact, rather than construct an actual classifier. This will instead be the purpose of Section 5, where we will use much more appropriate methods, methods that does not just myopically examine each data point in isolation, but also its context.

What we would want to see in the scatterplots is of course some evidence that the two classes of data points are separable. But we would also want to see inferred decision boundaries which are not parallel to either axis of the diagram. If this is the case, then one of the features contribute much more to the decision than the other. By contrast, diagonal decision boundaries would imply that both features are contributing to the decision. By inspection, we can see that the first case appears to be true for all feature pairs, except for those including GDC as one of the features. GDC appears to be largely irrelevant for the inferred decision boundaries.
Figure 4.2: Scatterplots of relationship between each pair of features. The data have been sampled from the dataset using non-parametric bootstrap with 100 samples in each diagram. Black lines
4.3. Relevance and usefulness of individual features

denote decision boundaries for support vector machines trained (10 trials) on similarly bootstrapped data (5000 samples).
4. Analysis

To analyze how much individual features contribute when using three or more features we now turn to feature selection. This is usually done using heuristics to greedily search for mutually promising features, but since we only have six features, the total number of combinations is just 64, and we can simply enumerate all possible choices and rank all of them. The results are given in Figure 4.4.

| $n$ | Feature selection | Binary classification accuracy |
|-----|-------------------|-------------------------------|
| 1   | \{μ\}             | 71.7% ± 2.6%                  |
|     | \{σ\}             | 68.9% ± 1.9%                  |
|     | \{Rank\}          | 61.9% ± 2.0%                  |
|     | \{tf · idf\}      | 60.4% ± 2.0%                  |
|     | \{γ\}             | 56.9% ± 2.4%                  |
|     | \{GDC\}           | 51.0% ± 1.0%                  |
| 2   | \{μ, σ\}          | 75.7% ± 2.5%                  |
|     | \{σ, tf · idf\}   | 74.8% ± 2.0%                  |
|     | \{μ, tf · idf\}   | 73.0% ± 3.5%                  |
|     | \{σ, γ\}          | 72.1% ± 2.8%                  |
|     | \{μ, γ\}          | 71.2% ± 2.3%                  |
|     | \{μ, GDC\}        | 71.2% ± 2.1%                  |
|     | \{σ, Rank\}       | 70.2% ± 2.4%                  |
|     | \{σ, GDC\}        | 69.4% ± 7.0%                  |
|     | \{μ, Rank\}       | 68.8% ± 3.6%                  |
|     | \{γ, tf · idf\}   | 63.3% ± 5.0%                  |
| 3   | \{μ, σ, tf · idf\} | 79.3% ± 1.7%                  |
|     | \{μ, σ, Rank\}    | 77.6% ± 1.9%                  |
|     | \{μ, σ, γ\}       | 76.3% ± 1.7%                  |
|     | \{σ, γ, tf · idf\} | 76.3% ± 2.9%                  |
|     | \{μ, σ, GDC\}     | 75.8% ± 2.2%                  |
|     | \{σ, tf · idf, Rank\} | 75.4% ± 2.6%                  |
|     | \{σ, γ, Rank\}    | 75.2% ± 1.6%                  |
|     | \{μ, γ, Rank\}    | 74.5% ± 1.2%                  |
|     | \{μ, tf · idf, Rank\} | 74.4% ± 5.7%                  |
|     | \{μ, γ, tf · idf\} | 74.4% ± 2.0%                  |
| 4   | \{μ, σ, γ, tf · idf\} | 81.4% ± 2.5%                  |
|     | \{μ, σ, tf · idf, Rank\} | 80.8% ± 2.1%                  |
|     | \{μ, σ, tf · idf, GDC\} | 80.4% ± 1.9%                  |
|     | \{μ, σ, γ, Rank\} | 79.1% ± 1.6%                  |
|     | \{μ, γ, tf · idf, Rank\} | 78.7% ± 1.6%                  |
|     | \{μ, tf · idf, Rank, GDC\} | 77.9% ± 2.0%                  |
|     | \{σ, γ, tf · idf, Rank\} | 77.8% ± 2.5%                  |
|     | \{μ, σ, Rank, GDC\} | 77.6% ± 2.6%                  |
|     | \{μ, σ, γ, GDC\}  | 77.1% ± 1.7%                  |
|     | \{σ, γ, tf · idf, GDC\} | 77.0% ± 3.1%                  |

Table 4.4: Performance for each subset of features of cardinality $n \in \{1, 2, 3, 4\}$ sorted by average accuracy. In the cases where the number of combinations exceed ten, only the top ten results for each $n$ are reported. Standard deviations are reported for each entry.
4.4. Named entities

The proposed features, the mean $\mu$ and standard deviation $\sigma$, are at the top of all lists. Furthermore, despite the very strong correlation between them, they are much stronger together than any other pair of features tested here.

It is a bit surprising perhaps, that the gdc, which is commonly held to be a strong indicator of collocation, and thus of commonly recurring phrases, give such poor results. It is not surprising that gdc does not work on its own of course. After all, cohesion measures such as gdc cannot distinguish between terms and commonly recurring fragments such as of the and similar, but gdc has been reported to work well in both technical term extraction (Vivaldi and Rodríguez, 2007), and in keyphrase extraction (Lopez and Romary, 2010) in concert with other features.

4.4 Named entities

We also consider the distributional properties of named entities.\footnote{Or rather their lexical representations.} These might seem conceptually distinct from technical terms, but named entities tend to behave much the same as technical terms. Being highly domain specific and generally denoting a single concept in the subject domain these can well be seen as technical terms. The difference lie primarily in what they denote: technical terms denote general concepts, whereas named entities denote specific instantiations of these concepts.

In this case we use the gold standard from the CoNLL 2003 NER shared task. Similarly to the technical terms and to the keyphrases we extract the compositional $n$-grams from the named entities and calculate the mean and standard deviations of the timelines. The named entities in the CoNLL dataset are divided into four classes: entities referring to living beings (per), entities referring to organizations (org), entities referring to places (loc), and entities that do not fit into any of the previous categories (misc). We plot the histograms for each of these separately in order to see if the different classes have different properties (Figure 4.3).

For the most part, the neological properties of the named entities look very much the same as the neological properties of technical terms and keyphrases (Figure 4.3). The major deviations are the per entities and the loc entities. In the case of the per entities, the separation is stronger, and in the case of the loc entities, weaker. The first result appears to be largely explained by the fact that the dataset is derived from the Reuters corpus, and thus consists of newswire. Consequently, many of the bigram named entities are of recently famous persons, such as George Bush and similar names which are very strongly neological. On the other hand, while new locations such as the Czech republic and Former Yugoslavia do appear occasionally, primarily due to political changes, this is comparatively uncommon compared to the appearance of new names of organizations or people.
Figure 4.3: Class separation between background terms, technical terms, and named entities extracted from the CoNLL dataset. The named entities have been separated into PER entities (top left), ORG entities (top right), LOC entities (bottom left) and MISC entities (bottom right). Histo-
Class separation for the means (bigram)

Class separation for the standard deviations (bigram)

Class separation for the means (unigram)

Class separation for the standard deviations (unigram)

Class separation for the means (bigram)

Class separation for the standard deviations (bigram)

Class separation for the means (unigram)

Class separation for the standard deviations (unigram)

gram bin size was set to 2 in all cases. The resulting histograms have been smoothed using Matlab's default settings (moving average with span 5), and normalized to sum to one in each case.
We will here detail the main results of the study. From a scientific point of view the results are chiefly of interest in our endeavor to support the hypothesis that neology is a strong indicator of technical terms, comparable or superior to other such features, but from an engineering point of view the results of course also give evidence that the system works as intended.

We will describe the results in two parts. First, we evaluate on a gold standard dataset for technical term extraction. Second, we evaluate on two application datasets where technical term extraction are commonly seen as a useful preprocessing step: keyphrase extraction and patent invalidation search.

5.1 Intrinsic Evaluation

To compare the different features with each other, we evaluate their performance individually using cross validation. Since the SemEval-2010 dataset that the gold standard is based on is split into 4 subject classes, it makes sense to use these four classes as the splits in the cross validation evaluation. By doing so we will implicitly test the ability of the algorithm to generalize from one domain to another. We also test the performance of the system using random splits, which will allow us to test higher numbers of splits, as well as to test the performance of the system when the training data and the testing data are of the same domain. This case is probably the most similar to how most authors have previously approached this testing.

We try three different types of tagging patterns: IO tagging, BIO tagging, and LIO tagging, although we will only test the performance different on different tagging schemes using 10 random splits. As we will see, the performance of the features differ between all these cases.

| I          | O          |
|------------|------------|
| Prec       | Rec        | F1         | Prec       | Rec        | F1         |
| Words      | 55.0%      | 10.8%      | 18.0%      | 85.1%      | 98.3%      | 91.2%      |
| Lowercased | 50.9%      | 15.5%      | 23.7%      | 85.8%      | 97.0%      | 91.1%      |
| Lemmas     | 54.6%      | 17.9%      | 27.0%      | 86.2%      | 97.1%      | 91.3%      |
| Word shape | 58.3%      | 31.4%      | 40.8%      | 87.9%      | 95.9%      | 91.7%      |
| Affixes    | 52.3%      | 29.0%      | 37.3%      | 88.1%      | 95.0%      | 91.4%      |
| POS        | 55.4%      | 36.9%      | 46.1%      | 92.1%      | 91.6%      | 91.9%      |
| Rank       | 70.2%      | 7.6%       | 13.7%      | 84.9%      | 99.3%      | 91.5%      |
| MI         | 24.2%      | 6.3%       | 10.1%      | 84.2%      | 95.5%      | 89.5%      |
| GDC        | 27.2%      | 8.3%       | 12.7%      | 84.4%      | 95.4%      | 89.6%      |
| tf-idf     | 25.3%      | 35.6%      | 29.6%      | 87.8%      | 80.7%      | 84.1%      |
| Neology    | 44.3%      | 34.2%      | 38.6%      | 88.9%      | 92.1%      | 90.5%      |

Table 5.1: Extraction performance when training the model on document from ACM subject category C2.4, Distributed Systems and testing on H3.3, Information Search and Retrieval.

Let us start by examining the performance when using one subject domain for training, and another one for testing (Figure 5.1). We will, both here and in the remaining sections,
5. Results

focus on the performance of the positive labels (i) rather than on the background (o), since we are mainly interested in the ability of the system to correctly identify technical term spans. In this particular case the result is generally consistent with the published literature, and the best single feature is the POS tags, by quite a large margin, although rank manages to get larger precision at the expense of much worse recall. Wordshape, affixes, and neology have lower, but still comparable performance. By comparison, MI, GDC, and rank exhibit very poor performance overall. In particular, the word oriented features, i.e. words, lowercased words, lemmas, word shape, and affixes, achieve high precision, but low recall, but the word shapes and the affixes achieve higher recall, presumably due to their ability to generalize better. Overall, we achieve higher precision than recall, except in the cases of POS tags and TF-IDF.

IO tagging, 4-fold cross validation with domain folds

|          | I          |         |         |         |         |         |         |         |         |         |         |         |
|----------|------------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
|          | P          | R       | F₁      | P       | R       | F₁      |         |         |         |         |         |         |
| Words    | 73.0%      | 22.4%   | 34.1%   | 85.5%   | 98.2%   | 91.4%   |         |         |         |         |         |         |
| Lowercased| 73.8%      | 26.0%   | 38.1%   | 86.3%   | 97.9%   | 91.7%   |         |         |         |         |         |         |
| Lemmas   | 69.2%      | 28.2%   | 39.7%   | 86.5%   | 97.4%   | 91.6%   |         |         |         |         |         |         |
| Word shape| 68.3%      | 26.9%   | 38.4%   | 86.1%   | 97.2%   | 91.3%   |         |         |         |         |         |         |
| Affixes  | 69.8%      | 41.8%   | 52.0%   | 89.3%   | 96.1%   | 92.5%   |         |         |         |         |         |         |
| POS      | 62.1%      | 44.7%   | 51.0%   | 89.4%   | 93.9%   | 91.5%   |         |         |         |         |         |         |
| Rank     | 67.3%      | 13.3%   | 21.8%   | 83.9%   | 98.2%   | 90.5%   |         |         |         |         |         |         |
| MI       | 21.9%      | 2.6%    | 4.7%    | 81.8%   | 97.3%   | 88.9%   |         |         |         |         |         |         |
| GDC      | 37.8%      | 3.8%    | 6.8%    | 82.2%   | 98.1%   | 89.4%   |         |         |         |         |         |         |
| tf-idf   | 54.3%      | 2.7%    | 4.9%    | 82.0%   | 99.0%   | 89.7%   |         |         |         |         |         |         |
| Neology  | 59.0%      | 30.4%   | 39.7%   | 87.4%   | 94.9%   | 90.9%   |         |         |         |         |         |         |

Table 5.2: Extraction performance using crossvalidation with folds according to the SemEval-2010 document categories.

We now evaluate the performance on the same data set using cross validation with the four subject categories as training folds (Figure 5.2) In other words, the first fold consists of subject category c2.4, Distributed Systems, the second fold of h3.3, Information Search and Retrieval, the third fold 12.11 Distributed Artificial Intelligence – Multiagent Systems and the fourth fold of 14 Social and Behavioral Sciences – Economics. The POS tags are again one of the most relevant features, but in this case the affixes exhibit similar performance. Words, lemmas, and lowercased words exhibit significantly better performance, whereas MI, GDC, and TF-IDF exhibit considerably worse performance compared to the previous test. Since the main difference between the previous experiment and this one is that we here use a larger training set as well as more training data, it seems plausible that the reason that the word form based features are more useful here is simply because the increased size of the dataset allows them to better capture the variation in word usage across domains. Put more simply, three domains are likely to collectively contain more of the words used in a forth domain than is just one of them. Thus, we can do technical term extraction simply by learning which words are commonly used in technical terms, but we should be prepared to collect a very large amount of training data before such methods will start becoming useful.
We now move on to using 10-fold cross validation using random folds, in which case both the training and test sets will contain a mixture of the domains. Consequently, we do not evaluate the ability of the features to generalize to new domains, but simply their ability to label terms using training and test data from the same domain.

Unsurprisingly, the word based features, words, lowercased words, lemmas and the affixes exhibit a marked increase in performance, although word shape does not. Similarly, affixes, rank, and neology also exhibit increased performance. A more surprising result is that both tf-idf and gdc exhibit significantly better performance, both in terms of precision and recall. Similarly, mi also exhibit much higher precision, but is still hampered by very low recall. We might take this to mean that these features are indeed indicative of technical terms, but that their distribution of values do not generalize to new domains.

Why are tf-idf and gdc unable to generalize to new domains? We might hypothesize that the term cohesion and term frequency of usage typical of technical terms differ from domain to domain. In other words, while term cohesion and term frequency of usage for technical terms differ from ordinary terms, the exact distributional properties varies from domain to domain, perhaps not very much, but at least markedly enough that the algorithm overfits the data and thus fails generalize to unseen data.

When we consider BIO-tagging we interestingly get better performance for the I-tags than in the IO tagging case, but unsurprisingly, we get overall worse performance for the B-tags. Otherwise there is little remarkable in these results.

Compared to the BIO tagging we now get higher performance, but more strikingly, we achieve more similar results for the L-tags and the I-tags than we achieved for the B-tags and the I-tags. This appears to be particularly true for affixes.
5. Results

|          | B     | I     | O     |
|----------|-------|-------|-------|
|          | P     | R     | F₁    | P     | R     | F₁    | P     | R     | F₁    |
| Words    | 71.4% | 29.2% | 41.4% | 70.4% | 38.3% | 49.4% | 87.6% | 98.0% | 92.5% |
| Lowercased| 73.5% | 35.1% | 47.5% | 72.3% | 47.8% | 57.5% | 89.0% | 97.9% | 93.2% |
| Lemmas   | 70.4% | 35.6% | 47.2% | 70.3% | 47.9% | 56.8% | 89.0% | 97.4% | 93.0% |
| Word shape| 65.5% | 23.5% | 34.5% | 61.8% | 31.7% | 41.8% | 86.3% | 97.4% | 91.5% |
| Affixes  | 69.1% | 48.6% | 57.1% | 68.5% | 61.7% | 64.9% | 91.8% | 96.2% | 93.9% |
| POS      | 56.0% | 33.3% | 41.7% | 58.3% | 57.5% | 57.7% | 89.4% | 94.0% | 91.7% |
| Rank     | 62.1% | 17.6% | 27.3% | 55.1% | 20.4% | 29.6% | 85.1% | 98.1% | 91.1% |
| MI       | 64.6% | 4.1%  | 7.7%  | 51.8% | 4.1%  | 7.5%  | 82.3% | 99.5% | 90.1% |
| GDC      | 52.3% | 15.3% | 23.6% | 47.3% | 24.1% | 31.9% | 85.0% | 96.7% | 90.5% |
| tf-idf   | 57.7% | 19.8% | 29.4% | 46.9% | 23.0% | 30.8% | 85.2% | 96.5% | 90.5% |
| Neology  | 53.9% | 29.9% | 38.4% | 56.6% | 47.2% | 51.4% | 88.7% | 95.1% | 91.8% |

Table 5.4: Extraction performance using crossvalidation with 10 random folds.

In this case we also find that lowercased words actually outperform lemmas slightly. The inverse was true in previous tests, in particular Figure 5.1. That we get better performance when using a larger training set should not be surprising, after all we should expect lemmas to generalize better than words when the training set is small, but when the training set is increased, this advantage disappears.

|          | B     | I     | O     |
|----------|-------|-------|-------|
|          | P     | R     | F₁    | P     | R     | F₁    | P     | R     | F₁    |
| Words    | 80.2% | 34.1% | 47.8% | 70.3% | 40.4% | 51.1% | 87.9% | 98.0% | 92.6% |
| Lowercased| 82.0% | 39.5% | 53.2% | 72.1% | 49.2% | 58.4% | 89.2% | 97.8% | 93.3% |
| Lemmas   | 77.2% | 38.9% | 51.6% | 69.1% | 47.0% | 55.7% | 88.9% | 97.3% | 92.9% |
| Word shape| 67.2% | 24.2% | 35.5% | 61.9% | 32.2% | 42.3% | 86.3% | 97.4% | 91.5% |
| Affixes  | 75.4% | 53.0% | 62.2% | 68.3% | 61.2% | 64.5% | 91.8% | 96.2% | 94.0% |
| POS      | 63.4% | 37.5% | 47.0% | 58.5% | 56.9% | 57.5% | 89.3% | 94.1% | 91.7% |
| Rank     | 73.5% | 20.5% | 31.0% | 57.1% | 17.6% | 26.6% | 85.0% | 98.6% | 91.3% |
| MI       | 64.6% | 4.3%  | 8.1%  | 55.3% | 4.5%  | 8.3%  | 82.3% | 99.5% | 90.1% |
| GDC      | 69.5% | 22.7% | 34.1% | 55.4% | 29.3% | 38.3% | 85.7% | 97.0% | 91.0% |
| tf-idf   | 52.9% | 18.0% | 26.8% | 45.3% | 25.7% | 32.6% | 85.6% | 96.3% | 90.6% |
| Neology  | 63.5% | 33.5% | 43.8% | 55.6% | 44.7% | 49.5% | 88.5% | 95.4% | 91.8% |

Table 5.5: Extraction performance using crossvalidation with 10 random folds.

LIO tagging, hard matching

The general results from the previous tests are roughly reflected in the results we get when we evaluate using hard matches, instead of evaluating on token labelling accuracy as we did previously. Overall, the performance drops about 10 percentage points. The exceptions are the GDC, which drops around 20 percentage points when using all the data, the TF-IDF, which drops around 30 percentage points in both cases, and the POS tags, which yield about the same results for hard matching as for token labelling accuracy. That POS tags are reliably able to
5.2 Extrinsic Evaluation

SincethedatasetweuseisnotwidelyusedandacceptedbytheNLPcommunity,we shouldtake some time to verify that the labelling is of appropriate quality. Wecould do this by manually examining the labelling, but there is a simpler and more objective way to test the quality of the labelling – using extrinsic evaluation of the trained technical term extractor. This also yield a method to verify that the technical term extractor produces reasonable and intuitive results.

5.2.1 Keyphrase Extraction

The SemEval-2010 keyphrase extraction task is well known as a gold standard for keyphrase extraction, which will be the aim of our first extrinsic evaluation.

We will here use our technical term extraction system to generate keyphrase candidates, and then use TF-IDF to rank the candidates. Thus, the only difference between our system

|                | All | C+H |
|----------------|-----|-----|
| Words          | P   | R   | F₁  | P   | R   | F₁  |
| Words          | 63.1% | 25.8% | 36.6% | 49.4% | 11.0% | 17.9% |
| Lowercased     | 65.5% | 34.8% | 45.5% | 48.1% | 14.9% | 22.8% |
| Lemmas         | 61.5% | 33.7% | 43.6% | 47.6% | 17.1% | 25.1% |
| Word shape     | 53.0% | 23.6% | 32.7% | 45.3% | 22.6% | 30.2% |
| Affixes        | 58.5% | 47.7% | 52.6% | 50.1% | 27.6% | 35.6% |
| POS            | 46.4% | 46.3% | 46.3% | 48.1% | 52.9% | 50.4% |
| Rank           | 47.6% | 12.6% | 20.0% | 51.2% | 6.7% | 11.8% |
| MI             | 22.7% | 3.7% | 6.3% | 18.6% | 5.3% | 8.2% |
| GDC            | 20.8% | 4.3% | 7.1% | 18.8% | 8.0% | 11.2% |
| tf-idf         | 23.7% | 6.7% | 10.4% | 10.5% | 14.5% | 12.2% |
| Neology        | 45.0% | 34.1% | 38.8% | 34.7% | 25.7% | 29.5% |

Table 5.6: Extraction performance using hard matching when 1) training the model on odd numbered documents and testing it on even numbered ones (left column) and 2) training the model on documents from the ACM subject category c2.4, Distributed Systems and testing on h3.3, Information Search and Retrieval (right column).

detect complete term spans should come as no surprise, since technical term span boundaries generally coincide with noun phrase boundaries.

General observations

Neology outperforms the other unsupervised features (MI, GCD, and TF-IDF) in all tests. When the domain of the training and test set differ, the difference in performance is quite large. Neology appears to have similar performance to lemmas and lowercased words, but appears to reach its maximum performance using less training data (Figures 5.1 and 5.2). Neology does not outperform POS tags in any of the tests. It is similarly outperformed by affixes except when the training set is very small. On the other hand it outperforms rank and wordshape on all tests, except in the first test (Figure 5.1), where it is outperformed by word shape.
and the baseline is the way the generate candidates, and a reasonably accurate technical term extractor should presumably improve the results over the baseline.

Unfortunately, the SemEval-2010 task description omits a lot of details about the baseline implementation, such as how candidates were extracted and whether plural and singular forms of nouns are considered different keyphrases or simply variants of the same keyphrase (Kim et al., 2010). The assumptions we make here is that the candidates are all \textit{n}-grams, and that plural and singular forms are considered different keyphrases, because the results then matches the results reported in the task description. In particular, the singular and plural of a noun should most likely be considered variants of each other in a production system. However, the results of both our system and the baseline drop about the same amount, roughly one percent, and this difference should thus not invalidate our main result in this evaluation.

| Precision | P@5 | P@10 | P@15 |
|-----------|-----|------|------|
| Top score (HUMB) | 39.0% | 32.0% | 27.2% |
| Top score (WINGNUS) | 40.2% | 30.5% | 24.9% |
| Our system | 28.1% | 22.5% | 19.8% |
| Baseline (TF-IDF) | 17.8% | 13.9% | 11.6% |

Table 5.7: The keyphrase extraction results when using the technical term extractor implemented in this study for generating keyphrase candidates, instead of using simpler candidate generation methods such as using all \textit{n}-grams not beginning or ending in a stopword. The baseline performance and the top scores achieved in the SemEval-2010 keyphrase extraction task are given for comparison.

1. vote vector, fooling set, second vote vector, pairwise election, nondeterministic communication complexity, votes, voting rules, voting rule, deterministic communication complexity, communication protocol, vote $z_i$, vote vectors, communication complexity, pairwise elections, lower bounds

2. competitive ratio, price-volume model, algorithm, online algorithm, book model, fund manager, volume variability assumption, brokerage, volume distribution model, market, competitive ratios, limit orders, market order, financial markets, selling algorithm

3. mediator agent, agents, negotiator agents, preference model, negotiation protocol, subgame perfect equilibrium, agent, pathology, mediation service, multi-issue negotiation, agentification, linear constraints, multi-issue problems, multi-criteria methodology, agent recommendations

4. query-document term mismatch, query terms, term mismatch, test collections, test collection, query expansion, querydocument term mismatch, query expansion techniques, query term, information retrieval, retrieval systems, conceptual relevancy, term removal, document collection, average precision

Table 5.8: Sample output (top 15 candidates) for four documents in the SemEval-2010 dataset. Correct keyphrases have been bolded.

We give the results of the evaluation in Figure 5.7, and sample output in Figure 5.8. These results would place us somewhere in the middle of the rankings in the SemEval-2010 task,
specifically in the 9th, 12th, and 11th place respectively, out of 19 participants (not counting our system). Note however, that our method is not trained on keyphrases, and the actual keyphrase extraction step uses only unsupervised measures (TF-IDF). By contrast the majority of the systems entered into the SemEval evaluation were supervised methods.

Since the measure used for extracting keyphrases is the same in the baseline and in our system, it seems likely that the contribution of our approach lies more in identifying appropriate term spans, rather than differentiating keyphrases from non-keyphrases in the final extraction step.

The intent of this comparison is to argue that the labellings that the technical term extractor produces is reasonable from the point of view of keyphrase extraction. We do not claim that the ranking method we use here (TF-IDF) is reasonable to use in actual deployment. In order to construct a decent keyphrase extractor we should use more advanced methods, such as those used by the top performing systems. What we do show is that better candidate generation can and do improve the performance in keyphrase extraction. We here only use a very simply ranking procedure and it is possible that more advanced ranking procedures would benefit less from a better pruned candidate list. Evaluating a system that uses technical term extraction in conjunction with more advanced ranking, however, falls well beyond the scope of this study.¹

Compared with the baseline, our system appears to output a greater number of multi-word terms (Figure 5.9), and it seems our output more closely resembles the distribution of lengths in the ground truth data, although the ground truth distribution seems to have a longer tail.

| n  | Our system | Baseline | Ground truth |
|----|------------|----------|--------------|
| 8  | 0          | 0        | 1            |
| 7  | 0          | 0        | 2            |
| 6  | 1          | 0        | 13           |
| 5  | 1          | 5        | 20           |
| 4  | 26         | 17       | 97           |
| 3  | 267        | 117      | 495          |
| 2  | 1394       | 656      | 1188         |
| 1  | 456        | 1350     | 449          |

Table 5.9: The number of terms of length n extracted by our system versus the baseline system, as well as the equivalent number in the dataset ground truth.

### 5.2.2 Invalidation search

The dataset we use here is the NTCIR-6 patent retrieval task dataset, and we use the English part of the dataset, which consists of all the patents granted by the United States Patent and Trademark Office during the period 1993–2002. The task set out in the evaluation is invalidation search, in other words, finding all the patent applications filed prior to a given patent’s filing date that might invalidate the claim in the newly filed patent. As such, we are searching for relevant documents, and are thus dealing with a fairly straightforward information retrieval task, except that the requirements of its users are very different from the requirements of the

¹For one, these systems are quite complex and uses a multitude of information. They also necessarily omit a lot of details, in order to fit the page limit for submission. Taken together, this means that it is unfortunately quite difficult to recreate these systems to any reasonable degree of approximation.
average Google user. Recall is here much more important than precision, since missing an invalidating prior patent is much worse than having to manually sift through long search results.

However, since we are simply interested in corroborating the appropriateness of the labelling in the Chaimongkol-Aizawa dataset, we do not need to actually perform information retrieval, which would be extremely expensive to perform on this dataset, especially since we intend to use fairly expensive natural language processing. We can much more simply test whether the technical terms in the gold standard patents overlap significantly with those patent files with which they are relevant. For the following experiment, we restrict ourselves to using the abstracts.

| m | n | p | Examples |
|---|---|---|----------|
| 9 | 1 | 52.9% | authorized nodes, destination packets, logical networks, multi destination packets, net server, nodes authorized, packet originating, virtual LAN, virtual networks |
| 8 | 2 | 34.8% | DSP pool, DTMF detection, LAN hub, TDM frames, buffer frame, multi protocol, packet networks, telecommunication devices |
| 7 | 3 | 50.7% | decoder processing, macroblock as encoded, pixel encoded macroblock, pixel encoding, pixel motion estimation, pixel pseudo macroblock, quantifies distortions |
| 6 | 7 | 48.7% | controlled device, controlled devices, expansion module, interface module, processor module, processor module segment |
| 5 | 11 | 57.3% | gate insulating, passivation layer, pixel electrode, storage capacitor, interface module, processor module, processor module segment |
| 4 | 8 | 49.3% | bivalent hetero, decolorization accelerating, donating coloring, electron accepting compound |
| 3 | 24 | 41.2% | conductive layer, conductive polymer, semiconductor components |
| 2 | 65 | 33.1% | Microfluidic devices, microfluidic device |
| 1 | 246 | 24.7% | atrial fibrillation |
| 0 | 633 | | |

Table 5.10: Results of the invalidation search evaluation. Each row denotes the number of documents n that have m technical terms in common with their relevant prior art. The column p signifies the percentage of technical terms in the abstract that m represents, averaged over all n documents.

We give the results in Figure 5.10. In the results, we have a number of patents whose terminology overlaps considerably with their relevant prior art, but we also have a very large portion which have one or zero technical terms in common with their relevant prior art. Overall, the technical term extractor seems to extract a surprisingly small number of technical terms from the patents in the dataset, in many cases only a handful. This might be due to a number of reasons, but it is likely to be partly because the patents generally use different language from the SemEval-2010 dataset, and thus we should expect some loss in labelling accuracy. For instance, they generally use multi-word terms much less frequently, particularly in the abstract. It should also be remarked that the abstracts are generally quite short.

It also seems that, in general, the documents that are mutually relevant do not generally coincide with the technical terms in the texts, but rather in 1) trivial background language, and 2) descriptions. Thus, patents and their relevant prior art seem to coincide not so much in terms of what they are about, but more in terms of what they say about the topic. We might
5.2. Extrinsic Evaluation

Thus be better off searching based on the aspect of the text, rather than its theme, as has been suggested by Weinberg (1988).

In the case of the NTCIR-6 patent retrieval dataset, the set of relevant documents for each query (topic) is given by the authors of the patent application, thus necessitating that the authors knew about that particular prior art. However, we should perhaps assume that in general the authors are not already aware of the relevant prior art, and that this kind of document is the one we should be the most interested in finding.

In a case where the relevant prior was not given by the applicants themselves, we are likely to have a selection bias such that the patent applications that are filed do not share terminology with their relevant prior art. On the one hand, if the patent applicant is unaware of the relevant prior art, they are less likely to use the same terminology. Conversely, if a patent applicant is unaware of the existing terminology, they are less likely to be able to find relevant prior art. Furthermore, weak patent applications – i.e. those for which there exist relevant prior art – would not gain anything by using the same terminology as the prior art, since this would make it easier to find relevant prior art that would invalidate the patent claim.

5.2.3 General remarks

We have extrinsically tested the trustworthiness of the labellings in the Chaimongkol-Aizawa dataset using two hypotheses, the first of which appearing to hold, the second of which appearing to hold to a much lesser degree.

We should note that, while the datasets generally consists of the same or similar kinds of documents, the nature of the two document collections are very different, and in particular the authors in the two document collections have very different rationale for choice in terminology. In the scientific literature that comprises the SemEval-2010 dataset we might assume that there is a strong tendency towards standardization of terminology. Scientists might make up new terms if they are unaware of existing terms for the concepts they wish to express, but in general they would want to use terminology that their peers would know and recognize. The overarching goal for the individual academic is after all the recognition of their peers, and making it harder for other people to find their work will only hurt themselves. By contrast, in patent applications, we might assume that there is less tendency towards standardization of terminology, since deliberately making it harder to find prior art actually improves a patent application’s chances of being granted. On the other hand, the hypothetical equivalent in an academic setting, obfuscating a research paper by deliberately using new terminology, does not seem likely to improve a paper’s chances of passing peer review in high quality journals.
The extraction of technical terms fundamentally depends upon the assumption that these are biased in certain ways. For instance, a common assumption taken in keyphrase extraction is that keyphrases are biased to occur more frequently at certain positions in the document. Another common assumption is that technical terms are biased to occur with different frequencies in certain communities, or that the contexts in which keyphrases and technical terms appear differ between different communities. Similarly to the position and community bias of technical terms, we suggest that technical terms also have a time bias, in that the technical terms of a document are skewed to be overrepresented in the contemporary and subsequent literature, but likely to be absent or severely underrepresented in the precedent literature.

The novel contributions of this work are as follows:

1. We have shown that technical terms tend to be recently coined, and that this statistical tendency is strong enough that it allows us to extract technical terms with reasonable accuracy.

2. We have introduced two measures of surface level neology that are simple and robust against noise, and that can effectively separate technical terms from non-technical terms.

3. We have shown that keyphrases have similar distributional properties as technical terms in terms of neology.

4. We have shown that named entities have similar distributional properties as technical terms in terms of neology.

The results are promising in that our proposed features appear to be useful for technical term extraction, which is of course the aim of this study, but they remain unable to improve upon the state of the art when used in supervised machine learning. They do however appear to improve upon the state of the art when compared against numerical features that can be used in unsupervised technical term extraction. As such these features are probably more reasonably seen as term cohesion measures, and seem to be more useful as such, when compared against similar such measures, e.g. GDC. While unsupervised extraction of technical terms using neology is not explored in this study, we have shown that supervised methods using our proposed features are able to achieve good performance even when trained on small amounts of data, and are able to generalize well to new domains.

Compared to others features however, the features we propose requires a fairly large amount of data to store. We can of course reduce the storage space requirements by preprocessing the data, but we will still require more storage space than we would for features like TF-IDF that are calculated directly on the documents that we search through.

The hypothesis in this study is that technical terms are neologisms, and we investigate this hypothesis using evaluation datasets that are heavily skewed towards computer science literature, where the terminology is by definition heavily neological. One might thus argue that this study presupposes its own conclusion by cherry-picking datasets where the hypothesis is trivially true. We should probably be prepared to see different distributions if we were to repeat the evaluations in Section 4 using datasets from domains such as literature, architecture, or law, where the terminology is stable, and does not change much over time. However, in subject
6. Conclusions

fields where the terminology is stable there generally exist hand-crafted terminologies, which by virtue of being of terminologically stable domains do not need to be frequently updated since the terminology is unlikely to become outdated. By contrast, it is precisely in those fields where the terminology changes quickly where we need automatic methods of extraction. Similarly, in terminologically stable fields it for the entries that are being introduced that we need automatic methods of extraction.

The very high values of the means, and the very low values of the standard deviations observed for the technical terms in Section 4 suggests that the majority of the technical term bi-grams under consideration here come from terms that only appeared after 1950. This might be explained by the fact that the datasets we use here are derived from the SemEval-2010 dataset, which is strongly biased towards computer science literature. It seems reasonable that the separation between the classes should generally be stronger in subject fields where the terminology tends to be very recent coinage than in fields with more mature terminology. If this is true, then the approach we propose here should work well for subject fields where the terminology is rapidly changing, and where the need for automatic extraction methods is arguably the greatest.
Aker, A., Paramita, M. L., Barker, E., and Gaizauskas, R. (2014). Bootstrapping Term Extractors for Multiple Languages. *Proceedings of the 9th edition of the Language Resources and Evaluation Conference*, pages 483—489.

Badia, T., Pujol, M., Tuells, A., Vivaldi, J., Yzaguirre, L. D., and Cabrè, M. T. (1998). IULA’s LSP Multilingual Corpus: compilation and processing. *LREC 1998*, pages 29–31.

Bauer, L. (1983). *English Word-Formation*. Cambridge University Press.

Bodenreider, O. (2004). The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic acids research*, 32(July 2003):D267–D270.

Boguraev, B. and Kennedy, C. (1999). Applications of term identification technology: domain description and content characterisation. *Natural Language Engineering*, 5(1):17–44.

Bougouin, A. and Boudin, F. (2013). TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction. *Proc. IJCNLP 2013*, pages 543–551.

Bourigault, D. (1994). *Lexter: un Logiciel d’EXtraction de TERminologie: application à l’acquisition des connaissances à partir de textes*. PhD thesis, EHESS.

Bs 3669:1963 (1963). Recommendations for the selection, formation and definition of technical terms. Technical report, British Standards Institution.

Cabré Castellví, M. T., Estopà Bagot, R., and Vargas Sierra, C. (2012). Neology in specialized communication. *Terminology*, 18(2012):1–8.

Chaimongkol, P. and Aizawa, A. (2013). Utilizing LDA Clustering for Technical Term Extraction. *Proceedings of the Nineteenth Annual Meeting of the Association for Natural Language Processing, Nagoya*, pages 686–689.

Dagan, I. and Church, K. (1994). Termight: Identifying and translating technical terminology. *Proceedings of the 4th Conference on Applied Natural Language Processing (ANLP’94)*, pages 34–40.

Daille, B. (1996). Study and implementation of combined techniques for automatic extraction of terminology. *The Balancing Act Combining Symbolic and Statistical Approaches to Language*, 1:29–36.

Daille, B. (2005). Variations and application-oriented terminology engineering. *Terminology*, 11(1999):181–197.

Dredze, M., Wallach, H. M., Puller, D., and Pereira, F. (2008). Generating Summary Keywords for Emails Using Topics. *Proceedings of the 13th international conference on Intelligent user interfaces*.

El-Beltagy, S. R. and Rafea, A. (2009). KP-Miner: A keyphrase extraction system for English and Arabic documents. *Information Systems*, 34:132–144.
Frantzi, K., Ananiadou, S., and Mima, H. (2000). Automatic recognition of multi-word terms: The C-value/NC-value method. *International Journal on Digital Libraries*, 3:115–130.

Georgantopoulos, B. and Piperidis, S. (2000). A Hybrid Technique for Automatic Term Extraction. *Proceedings of the ACIDCA 2000 Conference*.

Goldberg, Y. and Orwant, J. (2013). A Dataset of Syntactic-Ngrams over Time from a Very Large Corpus of English Books. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 241–247.

Harris, J. (1979). Terminology change: Effect on index vocabularies. *Information Processing & Management*, 15(2):77–88.

Hasan, K. S. and Ng, V. (2014). Automatic Keyphrase Extraction : A Survey of the State of the Art. *Acl*, pages 1262–1273.

Heid, U. and Gojun, A. (2012). Term candidate extraction for terminography and CAT : an overview of TTC. *Proceedings of the 15th EURALEX International Congress*, pages 585–594.

Hulth, A. (2003a). Improved Automatic Keyword Extraction Given More Linguistic Knowledge. *EMNLP’03: Proceedings of the 2003 conference on Empirical Methods in Natural Language Processing*, pages 216–223.

Hulth, A. (2003b). Reducing False Positives by Expert Combination in Automatic Keyword Indexing. *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, pages 367–373.

Hulth, A., Karlgren, J., and Jonsson, A. (2006). Automatic keyword extraction using domain knowledge. *Computational Linguistics and Intelligent Text Processing*, pages 472–482.

ISO 10241-1:2011 (2011). Terminological entries in standards – Part 1: General requirements and examples of presentation. Technical report, International Organization for Standardization.

ISO 704:1968 (1968). Naming principles. Technical report, International Organization for Standardization.

ISO 704:2009 (2009). Terminology work – Principles and methods. Technical report, International Organization for Standardization.

Jacquemin, C. (2001). *Spotting and discovering terms through natural language processing*. MIT Press.

Justeson, J. S. and Katz, S. M. (1995). Technical terminology: some linguistic properties and an algorithm for identification in text.

Kageura, K. (2002). *The Dynamics of Terminology*. John Benjamins Publishing.

Kageura, K. and Umino, B. (1996). Methods of automatic term recognition: A review. *Terminology*, 3:259–289.
Kageura, K., Yoshioka, M., Tsuji, K., Yoshikane, F., Takeuchi, K., and Koyama, T. (1999). Evaluation of the term recognition task. *Proceedings of the 1st NTCIR workshop on research in Japanese text retrieval and term recognition.*, pages 42–49.

Kim, S. N., Medelyan, O., Kan, M.-Y., and Baldwin, T. (2010). Seameval-2010 task 5: Automatic keyphrase extraction from scientific articles. *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Lemmens, K. (2011). The slow dynamics of legal language: Festina lente? *Terminology, 17*:74–93.

Lin, Y., Michel, J.-b., Aiden, E. L., Orwant, J., Brockman, W., and Petrov, S. (2012). Syntactic Annotations for the Google Books Ngram Corpus. *Proc. of the Annual Meeting of the Association for Computational Linguistics*, pages 169–174.

Litvak, M. (2008). Graph-Based Keyword Extraction for Single-Document Summarization. *MMIES ’08 Proceedings of the Workshop on Multi-source Multilingual Information Extraction and Summarization*, pages 17–24.

Liu, F., Pennell, D., and Liu, Y. (2009). Unsupervised approaches for automatic keyword extraction using meeting transcripts. *The 2009 annual conference of the North American chapter of the association for computational linguistics*, pages 620–628.

Loginova, E., Gojun, A., Blancafort, H., Guégan, M., Gornostay, T., and Heid, U. (2012). Reference Lists for the Evaluation of Term Extraction Tools. *Proceedings of the 10th Conference on Terminology and Knowledge Engineering*, pages 177–192.

Lopez, P. and Romary, L. (2010). HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID. *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 248–251.

Manning, C. D. and Schütze, H. (2000). *Foundations of Natural Language Processing*. MIT Press.

Maynard, D. (2000). *Term Recognition Using Combined Knowledge Sources*. PhD thesis, Manchester Metropolitan University.

Medelyan, O., Frank, E., and Witten, I. H. (2009). Human-competitive tagging using automatic keyphrase extraction. *EMNLP ’09 Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 3:1318–1327.

Medelyan, O. and Witten, I. H. (2006). Thesaurus based automatic keyphrase indexing. *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 6–7.

Michel, J., Shen, Y. K., Aiden, A. P., Veres, A., Grey, M. K., The Google Books Team, Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., Pinker, S., Nowak, M. A., and Aiden, E. L. (2011). Quantitative analysis using millions of digitized books. *Science*, pages 176–182.

Mihalcea, R. and Tarau, P. (2004). TextRank: Bringing order into texts. *Proceedings of EMNLP*, 4:404–411.

Murphy, K. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
Nakagawa, H. and Mori, T. (2003). Automatic term recognition based on statistics of compound nouns and their components. *Terminology*, 9(1):201–219.

Nazar, R. and Cabré, M. T. (2012). Supervised Learning Algorithms Applied to Terminology Extraction. *Proceedings of the 10th Terminology and Knowledge Engineering Conference: New frontiers in the constructive symbiosis of terminology and knowledge engineering*, pages 209–217.

Nguyen, N. and Yunsong, G. (2007). Comparisons of sequence labeling algorithms and extensions. *Proceedings of the 24th international conference on Machine learning*.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The PageRank Citation Ranking. *World Wide Web Internet And Web Information Systems*, 5:1–17.

Pantel, P. and Lin, D. (2001). A Statistical Corpus-Based Term Extractor. *Proceedings of the 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence - Advances in Artificial Intelligence (AI 2001)*, pages 36—-46.

Pavel, S. (1993). Neology and Phraseology as Terminology-in-the-Making. *Terminology: Applications in interdisciplinary communication*, 21:34.

Pinnis, M., Ljubešić, N., Ștefănescu, D., Skadiņa, I., Tadić, M., and Gornostay, T. (2012). Term Extraction, Tagging, and Mapping Tools for Under-Resourced Languages. *Proceedings of the 10th Conference on Terminology and Knowledge Engineering*, pages 193–208.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155.

Rey, A. (1974). *Essai de définition du concept de néologisme*. *Actes du colloque international de terminologie*, pages 9–28.

Rey, A. (1979). La terminologie: noms et notions. *Presses Univ. de France*.

Rose, S., Engel, D., Cramer, N., and Cowley, W. (2010). Automatic keyword extraction from individual documents. *Text Mining*, pages 1–20.

Sager, J. C., Dungworth, D., and McDonald, P. F. (1980). *English special languages: principles and practice in science and technology*. John Benjamins Publishing Company.

Sang, E. F. T. K. and Veenstra, J. (1999). Representing Text Chunks. *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 173–179.

Sapir, E. (1921). *Language: An Introduction to the Study of Speech*. Harcourt, Brace & Company.

Stenetorp, P., Pyysalo, S., and Tsujii, J. (2011). SimSem: fast approximate string matching in relation to semantic category disambiguation. *BioNLP ’11*, pages 136–145.

Sutton, C. and McCallum, A. (2007). *An Introduction to Conditional Random Fields for Relational Learning*. In Getoor, L. and Taskar, B., editors, *Introduction to Statistical Relational Learning*. MIT Press.
Temmerman, R. (2000). Towards new ways of terminology description: the sociocognitive-
approach. John Benjamins Publishing.

Tomokiyo, T. and Hurst, M. (2003). A language model approach to keyphrase extraction.
Proceedings of the ACL 2003 workshop on Multiword expressions analysis, acquisition and
treatment, 18:33–40.

Turney, P. D. (1999). Learning Algorithms for Keyphrase Extraction. Information Retrieval, 2.

Vivaldi, J. and Rodríguez, H. (2007). Evaluation of terms and term extraction systems: a
practical approach. Terminology, 13:225–248.

Wan, X. and Xiao, J. (2008). Single Document Keyphrase Extraction Using Neighborhood
Knowledge. Science And Technology, 2:855–860.

Wang, R., Liu, W., and McDonald, C. (2014). How Preprocessing Affects Unsupervised
Keyphrase Extraction. Computational Linguistics and Intelligent Text Processing, pages 163–
176.

Weinberg, B. H. (1988). Why indexing fails the researcher. The Indexer, 16(1):3–6.

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., and Nevill-Manning, C. G. (1999). KEA:
Practical Automatic Keyphrase Extraction. Proceedings of the fourth ACM conference on
Digital libraries, page 9.
