Faculté des **Langues**
Université de Strasbourg

Master Technologie des Langues
Option Traitement Automatique des Langues

2018-2020

Computational Stylistics: A Study of Enjambment

Automatic Detection of Enjambment

Eulalie Monget

Master's thesis

Under the supervision of
Pablo Ruiz Fabo

Jury: Mme Delphine Bernhard, M. Pablo Ruiz Fabo, Mme Amalia Todirascu

Acknowledgments

I would like to thank the following people, without whom I would not have been able to complete this research, and without whom I would not have made it through my master's degree.

My supervisor, Dr. Pablo Ruiz Fabo, for his dedication, reactivity and unfailing support. His experience working with enjambment specifically and natural language processing in general was invaluable and his enthusiasm for the subject infectious. More than a supervisor, he was a mentor, and I can never thank him enough for all the time he invested working with me.

All my English-speaking friends for helping me out when I needed them.

Léonie and Myléna, my classmates and friends, for being my lifebelt in a city I knew nothing of, and preventing me from quitting before I could even start this work.

Morgane and Sophie, my dear English teacher friends, for their proofreading and putting up with my constant doubts and questions about the clarity of my writing.

Julien, my partner, without whom I would never have had the guts to dive in the world of natural language processing, for being a constant reminder to quit doubting and start acting.

Table of Contents

List of Figures

List of Tables

Introduction

From early on, researchers have striven to study the effects of enjambment, be it its stylistic effects or its relation to metrics. Yet, studies such as Parry's (1929) on the effect of enjambment in Homeric lines, Lawler's (1978) on enjambment as a device to emphasise sexual metaphors or Tourrette's (2004) on Faur de Pibrac's use of enjambment are limited to small samples of poems. Research on enjambment, including the works mentioned above, is usually limited to either a specific poet or a small sample of poems. Such studies would benefit from a tool capable of automatically detecting enjambment, and better yet, capable of providing the lexical context in which it occurs, since it would allow researchers to gather more evidence to support their claims. Therefore, automated textual analysis would help to address a larger sample and to get a more systematic empirical basis.

Automation was initially applied to text analysis for industrial tasks (customer service, recruitment). However, more recently, the computational analysis of humanities texts has become more prominent as an application of Natural Language Processing. Such applications of natural language processing fall in what is called Digital Humanities.

Digital Humanities is a field at the intersection of computational methods and the study of human society and culture, including literature. It has grown more and more popular over the course of the last decades and its usefulness regarding researches is not to be proven anymore: digitisation of large corpora enables large-scale study of linguistic phenomena or the evolution of a language thanks to tools like Sketch Engine or even thanks to natural language processing techniques.

Natural Language Processing is concerned with the understanding of natural languages by computers, that is languages spoken by humans, such as English or French, as opposed to languages spoken by machines, such as Python or C. It is considered as a subfield of Artificial Intelligence and ranges from "low-level" tasks like tokenisation (dividing of a text into smaller units) to "high level" tasks like distant reading (large-scale analysis of textual data, allowing, for instance, to study an author's style and useful for authorship attribution).

As its name suggests, computational stylistics applies computational methods to the study of literary stylistics. Work in this field includes automatic scansion of poetry (Agirrezabal, 2017) and detection of alliteration and other sound devices (Kao and Jurafsky, 2012) to name a few. It was applied in Mulholland and Quinn (2013) to study lyricists' style, based on features like their use of the first-person singular pronoun, vocabulary and passive voice, and ascertain whether or not a lyricist was suicidal. Automatic detection of enjambment falls in the field of computational stylistics as well.

Studies on enjambment stylistics can be facilitated by automating its detection thanks to Natural Language Processing. Work on the automatic detection of enjambment does not abound. Studies have been published for Spanish poetry, based on written input (Ruiz et al., 2017), and German poetry, based on audio input from poetry readings (Hussein et al., 2018). For English, Houston (2014) is developing a system focusing on the computational analysis of Victorian poetry, including a system for enjambment detection that uses punctuation and part-of-speech cues. Work covering other periods and techniques is thus still to be done.

Taking this into account, this thesis is dedicated to the development of a tool for the automatic detection of enjambment in English poetry.

The structure of the thesis is as follows. Section 1 describes what an enjambment is, some of its stylistic effects and provides a few typologies. Section 2 presents the existing work on automatic detection. Section 3 describes NLP tools useful for enjambment detection. Section 4 acts as a

bridge between the state of the art and the work carried out in the thesis: it introduces challenges for the creation of an automatic enjambment detection tool and for the corpora this requires, and outlines how the challenges were addressed in the thesis. It also introduces the thesis' intended contributions and related deliverables: tools and corpora. Section 5 is about the construction and annotation of the reference corpus for enjambment detection created in the thesis. Section 6 presents JaDe, the system developed in the course of this thesis and section 7 presents the results and performances of the system. Section 8 is a case study to help picture the usefulness of a tool like JaDe. Finally, section 9 concludes and addresses possible avenues for future work.

1. Enjambment

Already found in Homeric verse, enjambment is a distinctive trait of poetry, though not so widely used since it is found in only a relatively small percentage of poems. Although poets have disregarded many poetic traditions along the way, with blank then free verse, they kept using enjambment as a way to create rhythm, suspense or other stylistic effects.

In this section, definitions of enjambment are discussed, as well as some stylistic effects it creates and different typologies are provided.

1.1. Definition

Several definitions for enjambment have been proposed. We start with a definition from a general reference work, because the inadequacies in this definition will serve to highlight the difficulty of properly characterising the phenomenon. We will then move on to specialised definitions from linguistics-based poetry scholarship.

As a general reference work, the Merriam-Webster dictionary defines enjambment as "the running over of a sentence from line or couplet into another so that closely related words fall in different lines",[1] which could be paraphrased as "a sentence going on from one line to another, breaking apart closely related words".

According to this definition, enjambment occurs within a sentence, spreading it onto two or more lines. However, note that this is imperfect: one can argue that this can happen with or without enjambment, in lines terminated by commas for example.

In general descriptive terms, a sentence can be characterized with the following definition from the free Oxford dictionary[2] (OUP, 2020) as

> "a set of words that is complete in itself, typically containing a subject and predicate, conveying a statement, question, exclamation, or command, and consisting of a main clause and sometimes one or more subordinate clauses."

So in the example below, according to this definition, there is one and only one sentence, which is "*Out of the night that covers me, black as the Pit from pole to pole, I thank whatever gods may be for my unconquerable soul"*, where "*out of the night that covers me*" and "*black as the Pit from pole to pole*" are respectively a prepositional phrase and an appositive adjectival phrase and "*I thank whatever gods may be for my unconquerable soul*" is the main clause.

> "Out of the night that covers me,
>
> Black as the Pit from pole to pole,
>
> I thank whatever gods may be
>
> For my unconquerable soul."
>
> William Ernest Henley, "Invictus: The Unconquerable"

While it can be argued that few lines correspond to a full sentence, it can also be argued that this example fits the Merriam-Webster definition because "*black"* is related to "*night*" and they fall in two different lines.

As just suggested, a non-specialist definition such as Merriam-Webster's does not identify the

[1] https://www.merriam-webster.com/dictionary/enjambment
[2] https://www.lexico.com/definition/sentence

context in which enjambment occurs precisely enough, a sentence being too large a syntactic unit. Specialist works have tried to provide more accurate definitions. Harai Golomb defines it as

> "the occurrence of a line boundary at a point where the structure of the preversified text, for reasons of syntax, lingual meaning and/or literary interpretation, does not permit the oral execution or the aural imagination of a pausal structure" (Golomb:39, 1979),

where preversified means that the poem is no longer presented in verse-lines but as running text. This means that enjambment occurs when a line ends unexpectedly from a semantic or syntactic point of view, bringing about a pause that should not exist between the words.

This definition means that enjambment occurs within a phrase or closely related phrases and no longer within a sentence. Since in the previous example all but one line are terminated by commas or full stops, corresponding to a full phrase, the only instance of enjambment is *"I thank whatever gods may be/For my unconquerable soul"* as the verb *"thank"* is separated from the prepositional phrase completing it.[3]

Another way of tackling enjambment is, as Hollander suggests, to see it

> "as a kind of spectrum, along which we would arrange all the possible ways of terminating lines, considered not as boundaries or termini, but as the kinds of cutting into syntax which the slant-dash notation illustrates"[4] (Hollander:99, 1975)

This spectrum ranges from what he calls *"soft ended-lines"* (lines terminated by a punctuation mark, which are not considered as enjambment) to *"hard ended-lines"* (the last word of the line is actually divided in two, as in William Carlos William "The Red Wheelbarrow": *"a red wheel/barrow"*). Although it is less explicit than in Golomb's definition, the idea of an unnatural pause or cut between syntactic units is also found in Hollander's, so both can be seen as relevant regarding automatic detection of enjambment.

Besides the term *enjambment* itself, with its various possible definitions, it is relevant in enjambment studies to introduce the terms *end-stopped line* and *run-on line*. A run-on line is characterised by the lack of punctuation at the end of the line while an end-stopped line is terminated by a punctuation mark. A punctuation at line end means that there is no enjambment. However, as will be discussed throughout this work, the converse is not true: the absence of punctuation at the end does not automatically mean that there is an enjambment.

1.2. Stylistic effects

As a poetic device, enjambment gives rise to various stylistic effects. For instance, the pausal structure defined by Golomb leads, among other things, to syntactic ambiguities, leading the reader to several possible interpretations, like in William Carlos Williams' "The Right of Way":

> *"I saw a girl with one leg*
>
> *over the rail of a balco*ny"

It can be understood in at least two ways: is the persona seeing a one-legged girl or a girl with one leg on one side of the balcony and the other one on the other side? However, as pointed out in Koops van't Jagt et al. (2014), the ambiguity does not exist (or is, at least, not as strong) in the

[3] Following common convention in studies about poetry, a forward slash ("/") is used to indicate the end of a verse-line when the poem is quoted within a paragraph.

[4] For clarification, the term "slant-dash" refers to what is more commonly known as a forward slash; Hollander is referring to how marking line-ends with the slash highlights the syntactic units broken up (or not) by the line-end.

preversified form of the lines:

"I saw a girl [with one leg over the rail of a balcony]."

Just like alliteration or consonance, the pausal structure of enjambment can help reproduce a sound or a feeling, as in "Chanson d'automne" by Paul Verlaine, where the repeated pauses help reproduce the languor of the persona:

"Les sanglots longs
Des violons
De l'automne
Blessent mon coeur
D'une langueur
Monotone."

1.3. Types

Several typologies of enjambment coexist. The typologies presented below have been established according to criteria based on part of speech, syntactic dependencies or semantic completeness.

1.3.1. According to Golomb

He defines two types of enjambment: prospective and retrospective (Golomb:293, 1979).

It is prospective when the line is syntactically incomplete and must be completed by the following line:

"A thing of beauty is a joy for ever:
Its loveliness increases; it will **never**
Pass into nothingness; but still will **keep**
A bower quiet for us [...]"

John Keats, "Endymion"

There are several cases of enjambment here and all of them are prospective. At the end of each enjambed line, the reader is left with a question: what will the loveliness never do? What will it keep? The syntactic incompleteness of each line also creates a semantic incompleteness.

It is retrospective when the line is syntactically complete but the reader needs the following line to get the right interpretation:

"Life is a broken-winged **bird**

That cannot fly"

Langston Hughes, "Dreams"

If we consider only the first line, the sentence is syntactically complete, since the verb *"is"* has both its subject *"life"* and object *"broken-winged bird"* and the second line says something more about that bird.

1.3.2. According to Quilis

In his work, Quilis (1964) identifies contexts where a pause is unexpected. His definition of enjambment is rather close to that of Golomb, as they both emphasise the unnatural pause that

enjambment creates between strongly cohesive units. He proposes three types of enjambments for Spanish poetry: lexical, cross-clause and phrase-bounded.[5]

Because each type depends heavily on part of speech, such as verbs, determiners and so on, which exist in many languages including English, it is assumed here that they are relevant for other languages as well. Nonetheless, it is not excluded that some categories may not apply to English and that others that were not highlighted during his study may exist in other languages.

1.3.2.1. Lexical

The lexical type occurs when a word is spread onto two lines, which corresponds to Hollander's hardest enjambment.

> "so much depends
> upon
> a red **wheel**
> **barrow**
> glazed with **rain**
> **water**"
>
> William Carlos William, *Spring and all*, "The Red Wheelbarrow"

1.3.2.2. Cross-clause

The cross-clause type occurs between a noun and the relative clause modifying that noun.

> "Life is a broken-winged **bird**
> **That** cannot fly"
>
> Langston Hughes, "Dreams"

1.3.2.3. Phrase-bounded

The phrase-bounded type occurs when two strongly syntactically connected elements of the same phrase are spread onto two lines. This type can be divided into several subcategories.

French and English examples are provided in Appendix 1 to show that the typology can apply to other languages than Spanish.

Broken units	**Definition**	**Broken units**	**Definition**
NP/Adj	Break up of a noun and the adjective modifying it	Relword	Break up a lexical word and the grammatical word (determiner, conjunction...) related to it
PP/Adj	Break up of an adjective and the prepositional phrase modifying it	VP/Adv	Break up of a verb and the adverb modifying it
Adv/Adj	Break up of an adjective and the adverb modifying it	VP/cprep	Break up of a verb and the mandatory prepositional phrase modifying it

[5] Quilis' work is written in Spanish. The English name for each of his types are translations by Martínez Cantón et al. (2019). Lexical; cross-clause; phrase-bounded: léxico or tmesis; oracional; sirremático.

Broken units	Definition	Broken units	Definition
Adv/PP	Break up of an adverb and the prepositional phrase modifying it	V_chain	Break up of a verb and its auxiliary or modal
NP/PP	Break up of a noun and the relative prepositional phrase modifying it		

Table 1: Phrase-bounded types

1.3.3. According to Hussein et al.

Hussein et al. (2018) used the terms "soft" and "hard" enjambments, as in Hollander's spectrum. In their terms, soft enjambments occur when the noun phrase is separated from the verb it is the subject of[6], and the hard one occurs when elements from the same clause (DET/NOUN; VERB/DOBJ) or the morphemes of a word are separated.

1.3.4. According to Spang

Spang (1983) talks about expansion[7] when the subject or the object do not appear in the same line as their governing verb. Although it is not a kind of enjambment *per se* because the perceived effect of separating these elements is softer than between other elements of a phrase*,* this type is detected and tagged by ANJA (Automatic eNJambment Analysis), the tool developed by Ruiz et al. (2017) for Spanish poetry. It is also assumed that it is relevant for other languages.

"Je me **souviens**
Des jours anciens
Et je pleure"

Paul Verlaine, *Poèmes saturniens*, "Chant d'automne"

"And her dressing did **outbrave**
All the pride the fields then have."

Ben Jonson, *Underwoods*, "How he saw her"

Regarding the automatic detection of enjambment, the types described by Quilis and Spang are the most relevant, since they all depend on part of speech or dependencies. Tools to automatically annotate both part of speech and dependencies exist; such tools are discussed in section 3.

[6] This kind of soft enjambment is very close to what Spang defines as *enlace*.

[7] The original term is *enlace*. *Expansion* is the translation proposed by Martínez Cantón et al. (2019).

2. Existing work on automatic detection of enjambment

Few works on automatic detection of enjambment have been published yet. It is a quite recent field of study, as the first paper on the matter was published in 2014. As for now, tools exist for three languages: German, Spanish, and English (although the English study, as explained further on, only focuses on poetry from the Victorian era).

2.1. In German read-out poetry

Hussein et al.'s study (2018) is about detection of enjambment in read-out poetry. They used 69 poems from the *l*yrikline corpus.[8]

Since they were working with oral material, they had to align oral data and text data first thanks to a tool based on the Sail Align algorithm (Katsamanis et al., 2011) and the grammetric principles (Wesling, 1996). They then tagged the text data with the *Stanford parser* and the Stuttgart-Tübingen-TagSet (Schiller et al., n.d).

Their postulate was that enjambment can occur in three different ways in read-out poetry: readers ignore the enjambment; they stress the pause and if it is a soft one, it has no impact on the reading flow; they stress the pause, and if it is a hard one, it has an impact on the reading flow.

Their tool also shows that there is a slight difference in reading between poets from ex-RDA, who tend to stress the enjambment (75%) and those from ex-RFA, who don't (23,5%).

As of this writing, their tool says whether the enjambment is emphasised or not. They plan on improving it so as to make a distinction between soft and hard enjambments (see 1.2.2) and, beyond enjambment, an overall goal of their research is to identify rhythmical patterns in free verse poetry.

2.2. In Spanish poetry

Ruiz et al.'s (2017) study is about detection of enjambment in Spanish poetry. They collected a total of 4,087 sonnets (15th-19th centuries) from online sources. The corpus thus created (DISCO, Ruiz et al. 2018) is available on github.[9]

They developed ANJA (see Martínez Cantón et al., 2019), a tool that detects enjambments, assigning them a tag according to Quillis and Spang's types. The tool is available online, for a user-friendly experience, and as a command-line software.

The types are assigned through a set of rules based, mainly, on POS (Part Of Speech) tagging and dictionaries for closed-class words, such as conjunctions.

For example, if the last word of the line has a tag DET and the first word of the following line has a tag NN, it will be marked as *pb_relword*.

The case of expansion is dealt with thanks to dependency parsing: if the subject does not appear in the same line as the verb it is governed by, then the line-pair is marked as *ex_subj_verb*. The list of enjambment tags is available on their website.[10]

Future works include lexical characterization in case of enjambment so as to determine the lexical context of enjambed lines, which could be useful for studies such as Lawler's (1978). His paper is about enjambment as a device to emphasise sexual metaphors, so large-scale analysis of poems

[8] www.lyrikline.org
[9] https://github.com/pruizf/disco

[10] https://sites.google.com/site/spanishenjambment/

for enjambed lines annotated with lexical information could be useful to characterise contexts where such metaphors can be found.

2.3. In English poetry

Houston's (2014) paper describes her approach to automatic detection of enjambment in a corpus of poetry from the 19th century.

It deals with three different measures of the relation between the poetic line and the syntactic sentence: a line:sentence ratio; a spectrum marking degrees of enjambment based on different kinds of punctuation[11] and part of speech tagging.[12]

She proceeds to explain that these three measures, all three features of poetic style, can be used, with other features, "in classification experiments or as markers of historical change in poetic practice".

As she explains, the interest of automatic detection of enjambment resides in the improvement of

> "our understanding of enjambment as a feature of an individual poets' style; as a feature of particular poetic forms, themes, and genres; and as a feature of poetic discourse in particular historical periods."

According to her, computational analysis of literary phenomena allows us to have a better understanding of the historical and cultural function of poetic form.

Since her study focuses on the Victorian era while this thesis aims to study enjambment throughout a larger period of time and the approach contemplated here is different from hers, it is still relevant to work on another system of automatic detection for English poetry.

[11] Based on Hollander's definition of enjambment, see section 1.1.

[12] Meaning a combination of Hollander's definition and Quilis' typology.

3. NLP tools relevant for enjambment detection

Natural language processing (NLP) is a broad discipline, concerned with the comprehension of human languages by machines. It ranges from spell checking to sentiment analysis, a task aiming at identifying the emotions expressed in a text, to machine translation, whose goal is to make translations easily available to all so as to improve communication between people from all over the world. Sentiment analysis is often used by companies to improve their after-sales services and machine translation, as said, allows people speaking different languages to understand each other or helps improving access to various sources of information since large amounts of textual data can be processed relatively quickly with satisfactory results (depending on the languages).

Regarding research in linguistics and literature, NLP can be applied in a myriad of ways, such as automating scansion of poetry (Agirrezabal, 2017) or enjambment detection (Houston, 2014, Ruiz et al., 2017, Hussein et al. 2018), determining whether a lyricist will or has committed suicide (Mulholland and Quinn, 2013) or preserving endangered languages or dialects by developing tools such as a spell checker to encourage speakers to use these languages (Lorteau, 2016).

This section presents technologies and tools useful for enjambment detection. As explained earlier, some types of enjambment depend on part of speech and syntactic dependencies, and several tools exist for the tasks at hand: assigning a part of speech tag to each word and analysing the relationship between them. A few tools, including the ones used in this work, will be discussed.

3.1. Part Of Speech tagging

Part of speech tagging (POS) is one of the basic tasks of almost any NLP projects. Each token of the text is assigned a tag according to its grammatical class (see Figure 1). A token is the basic unit for NLP tasks: it corresponds to a string of characters usually delimited by spaces. Punctuation marks and integers are also considered as tokens.

Two major tagsets are used to tag English texts: the Penn Treebank tagset and the British National Corpus (BNC) tagset.

Mary	had	a	little	lamb
PROPN	VERB	DET	ADJ	NOUN

Figure 1: POS tagging

3.1.1. Penn Treebank tagset

It was born out of the Brown Corpus tagset, which is very extensive, during the Penn Treebank project, led by Marcus et al. (1993). They strove to reduce the number of tags, many of which being unique to specific lexical items, while maintaining a good degree of precision. For example, they explain that the Brown corpus has three different tags for pre-qualifiers (*quite*, *rather*, *such*), pre-quantifiers (*all*, *half*, *many*, *nary*) and *both*. In the Penn Treebank tagset, all of them are tagged as predeterminers (PDT). The second version offers 55 tags.

3.1.2. BNC tagset

It is a quite extensive tagset, with 61 tags (including punctuation tags) for the regular tagset and offers 30 more tags, coined ambiguity tags. These ambiguity tags are attributed according to the probabilities given by the CLAWS automatic tagger, which was used to tag the 100-million words

BNC. As explained on the Lancaster University website,[13]

> "the ambiguity tag AJ0-AV0 indicates that the choice between adjective (AJ0) and adverb (AV0) is left open, although the tagger has a preference for an adjective reading. The mirror tag, AV0-AJ0, again shows adjective-adverb ambiguity, but this time the more likely reading is the adverb."

Most taggers can be trained with whatever tagset suits a given project best, but Penn and BNC are the two main tagsets for English. Although most taggers seem to use the Penn Treebank tagset by default, it would be interesting to compare the enjambment detection results obtained using the Penn tags and using the BNC ones, including the ambiguity tags.

3.1.3. Dependency parsing

Dependency parsing consists in studying the syntactic dependencies, meaning relationships, between words in a given sentence. For instance, a subject and an object are both dependencies of a transitive verb.

Lucien Tesnière, who is seen as the father of dependency grammar, states that

> "tout mot qui fait partie d'une phrase cesse par lui-même d'être isolé comme dans le dictionnaire. Entre lui et ses voisins, l'esprit aperçoit des connexions, dont l'ensemble forme la charpente de la phrase"[14] (Tesnière, 1959)

These connections establish the dependencies between each word of a sentence, and each connection links a "governing" word (regent) and a word governed by that word (dependent). Figure 2 illustrates a dependency tree for the sentence "Mary had a little lamb". A dependent can be regent in turn. The main regent is usually the main verb of the sentence.

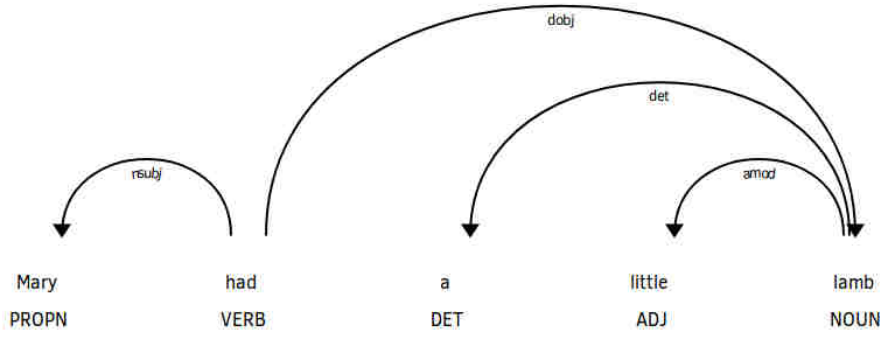

Figure 2: dependency parsing tree governed by "had"

Several NLP tasks benefit from dependency parsing. For enjambment detection, it is useful to detect a type called *expansion* (see 1.3.4).

Two major tagsets exist for dependencies in English: Stanford Dependencies and Universal Dependencies.

3.1.4. Stanford Dependencies

According to the 2016 manual (de Marneffe and Manning), the Stanford Dependencies contain around 50 types of dependencies, supposedly applicable cross-linguistically.

[13] http://ucrel.lancs.ac.uk/bnc2/bnc2guide.htm.

[14] any word that is part of a sentence is no longer isolated as it is in a dictionary. The human mind establishes connections between the words, the whole of which makes the structure of a sentence (*translation by Eulalie Monget).*

In their 2006 paper, de Marneffe et al. (2006) explain that several treebanks for dependencies surfaced in the early 21st century, including the Stanford one (Klein and Manning, 2003). The 2006 Stanford tagset, described in the same paper, is based on the dependencies defined by Caroll et al. (1999) and King et al. (2003), although their goal was to simplify these dependencies. They came up with 48 grammatical relations, many of which similar to those of Caroll, but that can be used more easily in relation or information extraction tasks.

3.1.5. Universal Dependencies

Universal Dependencies (UD) is an international collaboration project whose aim is to create treebanks of every language. It was born out of the merging of, among others, the universal Stanford dependencies and universal Google dependencies (Nivre et al., 2016).

The tagset for dependencies supported by UD lists 37 tags[15], which can be organized into dependencies relative to the core argument (governing word) or relative to a noun for exemple.

Regarding enjambement detection, the UD tags should be sufficient for detecting cases of expansion[16], as expansion occurs between a verb and the subject or the object of this verb. Tests with the Stanford tagset or other tagsets would of course also be interesting.

3.2. NLP libraries

Several libraries exist as a lot of programming languages, such as Perl, Python, Kotlin or Java, are used for NLP tasks. The library used in this thesis is spaCy, discussed below along with some others to provide a wider picture. Being knowledgeable mostly in Python and Java, only libraries for these languages are described in this section.

3.2.1. spaCy

SpaCy[17] is an open-source Python library offering support for over 48 languages. It claims to be the fastest in the world and provides pipelines for tokenisation, lemmatisation, POS tagging and dependency parsing among a much more comprehensive list of pipelines. It also offers the possibility to be trained with neural networks, and is compatible with TensorFlow, PyTorch and Scikit-learn.

Version 2 provides convolutional neural networks models for POS tagging, dependency parsing and named-entity recognition.

Being open-source, several non-native pipelines are made available by the spaCy community, such as the Berkeley Neural parser which is a constituency parser, spacy-wordnet which allows the use of WordNet and WordNet Domains or even a pipeline that detects emoji and generates a description. The Wordnet pipelines could be useful for lexical characterisation of the context in which an enjambment occurs and the Berkeley pipeline could be useful to disambiguate sentences.

Both the constituency parser and the Wordnet pipeline are not used in this thesis, but could be useful for further work.

3.2.2. NLTK

[15] http://universaldependencies.org/u/dep/index.html

[16] See p.7

[17] https://spacy.io

Natural Language ToolKit (NLTK)[18] is also an open-source Python project. It provides libraries for tokenisation, POS tagging, dependency parsing and stemming among others, as well as interfaces to 50 corpora and lexical resources, including WordNet. It is widely used by the NLP community, especially in teaching.

3.2.3. CoreNLP

Unlike the other two, Stanford CoreNLP runs using Java. It provides support for English, French, German, Chinese, Spanish and Arabic. It is quite comprehensive and offers services such as POS tagging, named-entity recognition, parsing (constituency and dependency) and sentiment analysis. The English tagger uses the Penn Treebank tagset. Manning, Christopher D., et al. (2014) present the tool in more depth.

3.3. Comparison

It is interesting to test and see the different components useful for enjambment detection. This section discusses some tests carried out with some sentences likely to be challenging for state-of-the-art NLP technologies, to get an indication of different pipelines' performance.

The tests were carried out with spaCy and CoreNLP. No test was done with NLTK dependency parser as dependency parsing in NLTK can be done with CoreNLP through an import.

The two sentences chosen for these example tests (below) are interesting to look at since they are either ambiguous (as in example 1) or rather representative of poetic aspects; example 2 disregards some written norms, particularly regarding punctuation cues.

Example 1: This dog from the street stinks and barks.

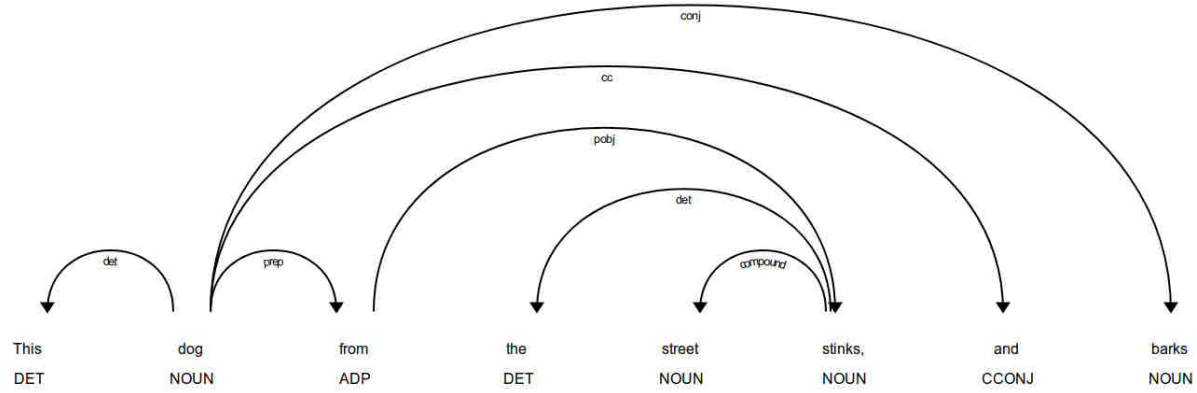

Figure 3: analysis of example 1 with spaCy

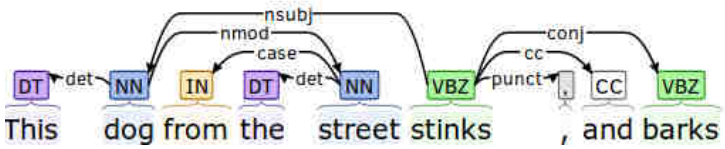

Figure 4: analysis of example 1 with CoreNLP (demo)

This example is troublesome for spaCy (Figure 3). Firstly, "*stinks*" and "*barks*" are mislabelled: they are tagged as NOUN whereas they should have a VBZ tag. Secondly, the dependencies are all wrong: The tree shows *conjunction* between "*dog"* and "*barks"* when it should show an nsubj

[18] http://www.nltk.org/

relation, and the relation between *"the"* and *"stinks"* is also wrong. There is actually no verb detected by the parser. On the other hand, CoreNLP (Figure 4) makes no mistake.

Example 2: "The old chronicles have much to say/Terrible omens speak of danger that lay ahead (Zenitram, "The gods tumble and fall" (from allpoetry.com))

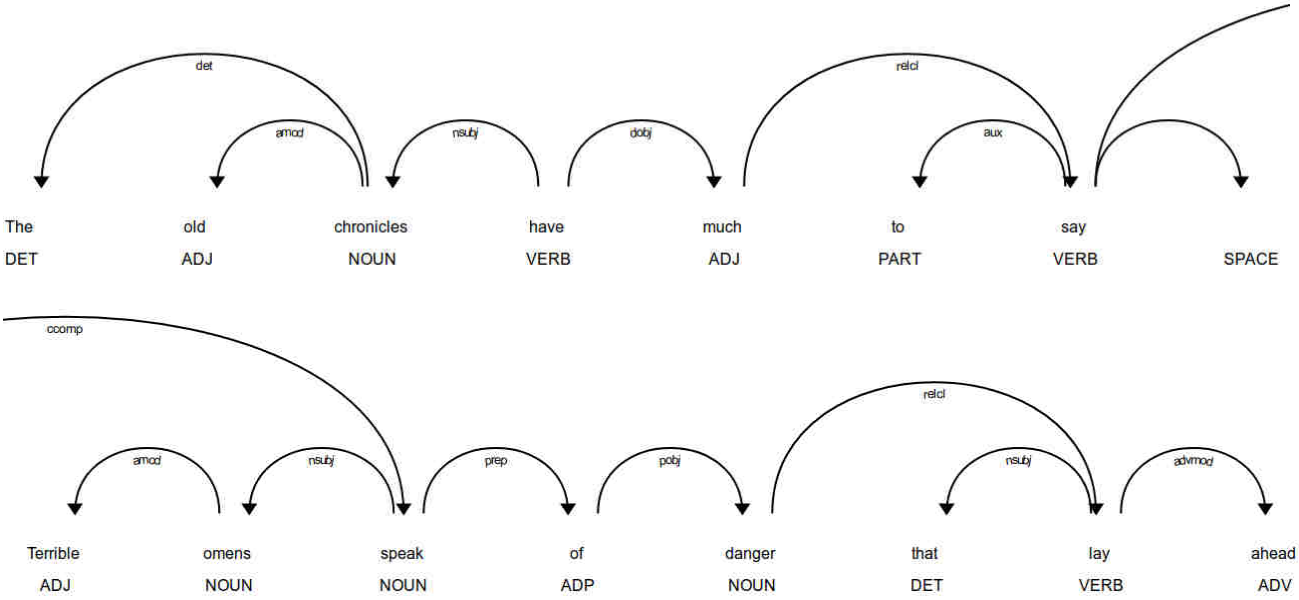

Figure 5: analysis of example 2 with spaCy

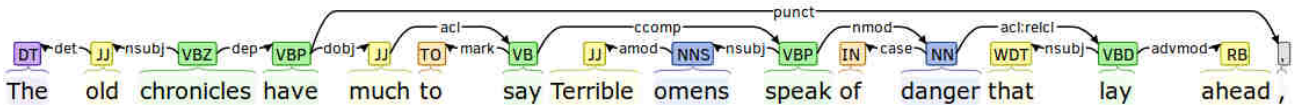

Figure 6: analysis of example 2 with CoreNLP

Neither of them gives a satisfactory output. Be it spaCy (Figure 5) or CoreNLP (Figure 6), both interpret "*Terrible omens speak of danger that lay ahead*" as a subordinate of "*The old chronicles have much to say*", which makes no sense since say either is a transitive or intransitive verb and, as such, cannot have two objects. Besides, an omen is not said:

?The old chronicles have to say much terrible omens [...]

The figure below, taken from the spaCy website, is based on a study by Choi et al (2015). The study compares 10 dependency parsers on criteria such as speed and accuracy. The corpus used for this comparison is the English sub-corpus from the OntoNotes 5 corpus, a large multi-lingual, multi-genre corpus annotated with syntactic structure, predicate-argument structure, word senses, named entities, and coreference (Weischedel et al., 2011; Pradhan et al., 2013). It shows that spaCy is slightly better than most of the tools it was compared with.

SYSTEM	YEAR	LANGUAGE	ACCURACY
spaCy v2.x	2017	Python / Cython	**92.6**
spaCy v1.x	2015	Python / Cython	91.8
ClearNLP	2015	Java	91.7
CoreNLP	2015	Java	89.6
MATE	2015	Java	92.5
Turbo	2015	C++	92.4

Figure 7: benchmark of several NLP libraries (© Choi et al., 2015) (accessed May 1st 2019)

The tools presented in this section were briefly compared in order to choose one for implementing the system developed in the thesis. Given its completeness, ease of use and comparable results to the others, spaCy was favoured and used in the thesis.

4. Towards a new system of enjambment detection in English: outline of the thesis' contributions

It can seem relatively easy to detect enjambment automatically, given that the basic aspects of the phenomenon can be captured with syntactic and punctuation cues. This is particularly true for poets playing by the rules of language and not of stylistics, since the language in poetry only observes the rules of language the poet wants to respect. Nevertheless, detecting enjambment can be more challenging than it would seem at first, and marking its type can be a challenge as well. This section discusses such challenges, which have served to shape the work developed in this thesis. It also outlines the intended contributions in the thesis regarding those challenges, including the deliverables produced (system and corpora).

The section structure is as follows: in 4.1, linguistic and stylistic challenges faced by the system are discussed. Section 4.2 justifies the need to adapt existing typologies to enjambment detection in English specifically. In 4.3, corpora created for the thesis are introduced.[19] Section 4.4 lists the thesis' intended contributions.

4.1. Difficulties for automatic detection

The typologies seen in section 1.3, especially those of Quilis and Spang, which rely respectively on POS tags and syntactic dependencies, can easily be used to automate enjambment detection. The ANJA detection system (Ruiz et al., 2017) attributes the tags according to the POS tag of the words directly before and after the enjambment, as well as based on the dependencies. Regarding English, such a system could meet several pitfalls, which will now be discussed.

4.1.1. Linguistic ambiguities

4.1.1.1. Independent phrases

As said, using only POS tags to determine the enjambment type is sometimes not enough. Indeed, it is possible that the enjambment breaks apart two different syntactic units of the sentence: types defined by Quilis and Spang are valid only within a phrase, so if enjambment occurs between an NP and VP (or a PP and a VP and so on) that are not related, it does not fall into any category described here.

For instance, in *"this dog from the street/barks and stinks"*, a human knows that a) the dog comes from and/or lives in the street and b) it is the dog that stinks and not the street. However, if a system were to only take POS into account, it is highly possible that it would consider that there is an expansion between *"street"* and *"stinks"* whereas dog is its real subject. This mistake could be avoided thanks to dependency parsing.

4.1.1.2. Phrasal vs (di)transitive verbs

Another difficulty that the system could meet concerns phrasal verbs. Phrasal verbs are quite vernacular in English and some verbs exist with or without a particle or a preposition.

1. You can count on me if you need help.
2. He's twenty and still counts on his fingers.
3. I can count on the fingers of one hand how many times he's been here.

These three instances of *count* work differently each time: in (1), "*count on"* is a phrasal verb, where "*count on someone*" means one can rely on someone else; in (2), "*on his fingers"* is just a

[19] Note that the manually annotated reference corpus is further described in a section of its own (section 5).

preposition acting as a complement of manner (how does he count? On his fingers); and in (3), "*count on the fingers" of one hand* is an idiomatic expression meaning that something does not occur often and is one united block. As a consequence, were they cases of enjambment, each of them should be annotated with a different type.

To avoid mislabelling cases like this one, it will be necessary to create a dictionary of phrasal verbs.

4.1.2. Stylistic liberties

Poetry is a form of art whose beauty resides in the poet's abilities to play with words, punctuation and linguistic norms. As a consequence, it is very likely to find occurrences of poems where line *n* is both syntactically and semantically complete but not terminated by any punctuation mark and line *n+1* is totally independent from *n*. In such cases, it cannot be considered as an enjambment, because *n* and *n+1* are two different sentences: there is no overflowing of sentence nor meaning, from one line to another. Consider the following example:

> "The old chronicles have much to say
>
> Terrible omens speak of danger that lay ahead," (sic)
>
> Zenitram, "The gods tumble and fall" (from allpoetry.com)

In this example, line *n* is syntactically complete: "*The old chronicles"* is the subject of "*have"* and "*much to say"* is the direct object; "*much"* acts as the object of "*say"*, so it is semantically complete as well. Regarding line *n+1*, "*terrible omens"* is the subject of "*speak of"* and "*danger that lay ahead"*, so here again, the structure subject-verb-complement is complete and there is no need for completion regarding meaning.

Yet, there is nothing to link or separate the two sentences: one could expect a colon ("*the old chronicles have much to say: terrible omens speak of danger that lay ahead"*), in which case *n* introduces *n+1*, or a comma ("*the old chronicles have much to say, the terrible omens speak of danger that lay ahead, (and...)"*), in which case *n* and *n+1* are in apposition.

Such an example is challenging for a machine, because semantic information is very important here. This is CoreNLP's analysis of the two lines:

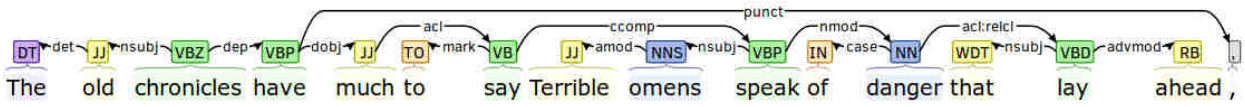

Figure 8: dependency parsing of a problematic example

It analyses *n+1* as a subordinate clause to verb *say* in line *n*, as if there were a subordinate conjunction, since it is possible in English to subordinate a clause to another one without an explicit conjunction *"that"*.

The lack of standard in punctuation cues and the non-standard uses of linguistic norms could be troublesome for dependency parsers.

4.2. Redefining a typology specific to English

As mentioned in 1.3.2, it is assumed that Quilis' (1964) and Spang's (1983) types are relevant for other languages than Spanish, since their types are respectively based on part of speech and dependencies. Yet, even though these features exist in a lot of languages, English and Spanish are quite different, the former being a West-Germanic language and the latter being a Romance

language. Therefore, it is possible that some types defined by Quilis during his experiment may not exist in English and that others not highlighted may, on the contrary, exist in English and not in Spanish.

One such example is the breaking apart of two adjectives modifying the same noun. In English, it is quite common to add up adjectives according to the opinion-size-age-shape-colour-material-origin-purpose order. A concrete example of the phenomenon applied to enjambment is the following:

> "I see you drinking at a fountain with **tiny**
> **blue** hands, no, your hands are not tiny,"
>
> Charles Bukowski, "An almost made up poem"

In the light of these differences, both typologies were adapted to better deal with English enjambment. Further details about this new typology are given in section 5.

4.3. Poetry corpora used

Finding appropriate poetry corpora was also an important task for this thesis. On the one hand, it was necessary to manually annotate enjambment in several poems, in order to create a test-corpus to evaluate the developed tool. On the other hand, in order to get an indication of the tool's potential for literary research, a large poetry corpus to which the tool could be applied was also needed.

Towards these goals, a corpus comprising sonnets and other forms of poems was collected. Using different forms of poetry is relevant since it may unveil different uses of enjambment according to a specific form. For the same reason, work from both male and female poets is included. The corpus is diachronic, including poems from the 16th century to the 21st, to account for the possible evolution of enjambment over time.

Including poems from both unknown poets and well-known poets could have been relevant as well regarding the purpose of the tool: it could possibly unveil new types of enjambment and be used to study the differences between successful poets and unknown ones, as in Kao and Jurafsky (2012). In that study, the style of both unknown and famous poets is analysed through computational methods (such as the automatic determining of the types of rhymes used or a tool to detect alliteration) in order to ascertain what makes an author successful.

The source for both the manually annotated corpus and the large application corpus was the website PoemHunter[20]. Despite the possible restriction of redistribution and the lack of metadata, it was the most versatile source among those we considered (described below), thus fitting best the criteria of various poetic forms and diachronic corpus.

A total of 9,749 poems from 62 authors from the 14th to the 21st centuries were collected. Most of these poems belong to the public domain. Out of these, 59 poems were manually annotated for enjambment and used as the reference corpus to evaluate the tool quantitatively (see section 5). A wider subset, with ca. 625 poems and ca. 25,000 lines was used for a case-study on our tool's automatic annotations, presented in section 8.

Note that, whereas the Poem Hunter corpus best covered our needs, it also presents some shortcomings. Other websites or databases were also examined as possible data sources, and could be used in the future in order to enrich the dataset. These sources are described below:

[20] https://poemhunter.com

- Literature Online (LION), a database dedicated to English literature. It gathers, according to their website, “over a third of a million full-text works of poetry, prose and drama in English”. The database lists more than 425,000 poems. Regarding sonnets, there would be over 10,000 poems available, from the mid-sixteenth century to the beginning of the twenty-first century. For each poem, metadata about the title, author, place and date of publication and more are given. Such metadata would be useful in order to analyse enjambment according to different corpus partitions. However, the corpus could not be directly made available for other research as the license granted by ProQuest, LION host, does not allow redistribution of content.
- Project Gutenberg,[21] a website making over 58,000 free ebooks available for download. It is hard to estimate the number of poems that could be retrieved from the website. When searching for sonnets, only 7 ebooks match. However, the metadata would have to be manually added to each poem, even if some can be found in the file. Most poems can be redistributed by users, making it possible to make the corpus available for anyone.
- Allpoetry.com,[22] a website dedicated to poetry. It offers a large number of poems, by canonised and non-canonised authors, but some poems are translations and not original works. Such poems could be a source of errors, for enjambment detection or for other research purposes, as a translation may introduce structures or figures of speech that do not exist in the original work, or on the contrary, remove such characteristics. Nonetheless, it is not devoid of interest to work on translations. Poems on this website are still owned by their original authors.
- Eighteenth-century Poetry Archive (ECPA),[23] a database comprising 2,920 works by 310 different authors. Each poem is encoded with the TEI standard and annotated with metadata and linguistic data (POS tag, if there are rhymes, metric of the line...). Every contribution submitted to ECPA is under a Creative Commons Attribution-NonCommercial-ShareAlike license, meaning they can be redistributed.

4.4. Intended contributions

Throughout 1.3, 4.1 and 4.2, different types of enjambment have been discussed and potential challenges for the definition of enjambment and its automatic detection have been highlighted. Based on these challenges, several tasks were defined in order to develop a new tool for enjambment detection, which this thesis aims to tackle. These tasks are outlined below, as well as the thesis’ contribution towards them. The remaining chapters of the thesis discuss the work carried out regarding those tasks:

- Defining the **enjambment typology** that the automatic detection system will tag against. The typology developed for this study, as well as a **guide for manually annotating** the reference corpus (annotation guide), can be seen in Appendix 2, or online at https://zenodo.org/record/3992703. See also Monget and Ruiz (2020).
- Creating a **reference corpus** for enjambment, annotated using the typology that will be defined in the course of this study. The reference corpus is described in section 5. Some example annotated poems are given in Appendix 3, and the complete corpus is available on GitHub, at https://github.com/MongetE/JaDe/tree/master/JaDe/resources/annotated_poems.

[21] www.gutenberg.org/

[22] https://allpoetry.com/

[23] https://www.eighteenthcenturypoetry.org/

- Coming up with and implementing a **set of rules** to detect and attribute a type to each enjambment, based on part of speech tags, syntactic dependencies and, if possible, constituency. The **system implemented** to apply these rules, besides the rules themselves, is publicly available on GitHub, at https://github.com/MongetE/JaDe, under a GNU General public license.
- As part of the detection system development effort, it appeared necessary to create or obtain a **dictionary of phrasal verbs**, to handle certain enjambment cases. The list created is among the system's resources on GitHub.[24]
- Providing a **case study** to show the potential contribution of the system developed to literary studies; this can be seen in section 8.
- In order to develop the case study, a **large digital corpus** of poems was necessary. The corpus source was just described in section 4.3 and the corpus created based on it (see section 8) is available on request.
- Finally, as a potential pedagogical resource, a **tutorial on visualizing** enjambment annotations for different authors was also created, as a Python notebook, and made available on GitHub. A stable html version is also available at https://nbviewer.jupyter.org/github/MongetE/JaDe/blob/master/JaDe/jane/jane.ipynb as well. The tutorial uses the charmak package, created in this thesis, to exploit the bokeh visualization library more easily.

[24] https://github.com/MongetE/JaDe/blob/master/JaDe/resources/phrasal_verbs.txt

5. Creating and annotating a test corpus for enjambment detection

5.1. Description

As seen in section 4.3, the PoemHunter website was used as our corpus source, since it provides a substantial amount of poems per author (hundreds of poems for some), covering a variety of origins and periods.[25] In total, more than 9,700 poems from 62 authors from the 14th to the 21st centuries were collected. Most of these poems belong to the public domain.

Based on the poems collected from that source, a corpus of 59 poems (for a total of 952 lines), 35 of which are sonnets (504 lines), was annotated by two annotators (the author and her supervisor, both familiar with English grammar and specificities).

Inter-annotator agreement was calculated with Cohen's kappa (Cohen, 1960), using the NLTK library's implementation.[26] Note first of all that the computation of inter-annotator agreement with kappa is a task that can be strongly affected by how we define the cases to consider as input to the agreement calculations, as Fort (2012: 164ff) explains. The values given here can just be seen as an orientation, our choices are made explicit for more transparency.

First, agreement on the presence vs. absence of enjambment was computed (all labels were neutralized to a single category to indicate the presence of enjambment vs. no annotation). A kappa of 0.81 indicated strong agreement regarding whether there is an enjambment or not in a given line-pair (ignoring the specific enjambment type entered by the annotators). Agreement for enjambment classification, on the specific enjambment types given by both annotators, was also calculated. This is more complicated because there are many more lines without enjambment than with it. Accordingly, if we take into account cases where both annotators consider that the line-pair has no enjambment for agreement, kappa will artificially increase. For that reason, cases where both annotators have no enjambment were ignored for agreement. Under these conditions, kappa was 0.63. This seems satisfactory for the present work.[27]

Given the small size of the corpus, it may not be truthfully representative of the phenomenon, but we consider it sufficient to provide a baseline version of the typology and/or the tool. Furthermore, the following diagrams on enjambment distribution cannot be used to give a representative insight of the phenomenon but are nonetheless relevant regarding our corpus.

[25] Other sources were also considered and could be used to enrich the dataset, see section 4.3.

[26] https://www.nltk.org/_modules/nltk/metrics/agreement.html

[27] If we accept cases where both annotators have no enjambment, kappa is 0.78, but as said this overestimates agreement.

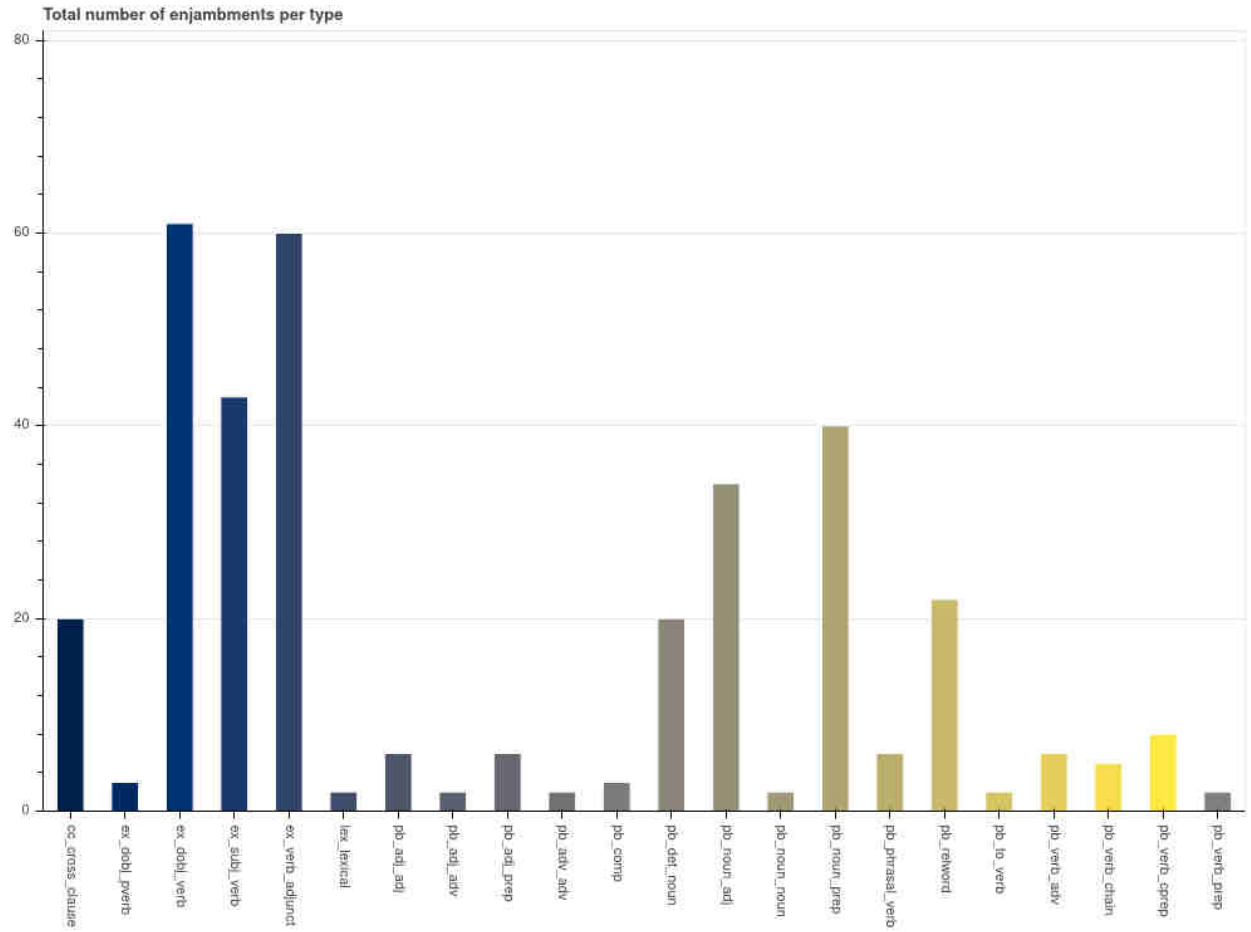

Figure 9: Overall distribution by types in the test corpus

The choice of sonnets is motivated by the strict set of rules governing their writing and thus by the homogeneity it provides. On the other hand, some of the poems chosen to be annotated are less conservative, so to speak, such as a few poems by Bukowski. Lines in these poems are indeed sometimes very short, reduced to only one word. It can be argued that these shorter lines could introduce types that have no reason to exist anywhere else, or that we cannot even talk about enjambment in such cases, because the reader will expect the phrases broken up by such short lines to continue in the next line anyway. However, no enjambment types were exclusively observed in these short-line contexts. Yet, a trend that was indeed attested was for the more recent poems in the corpus to introduce types not used in earlier periods, as shown by Figure 10 below (*pb_noun_noun, pb_comp* for instance).

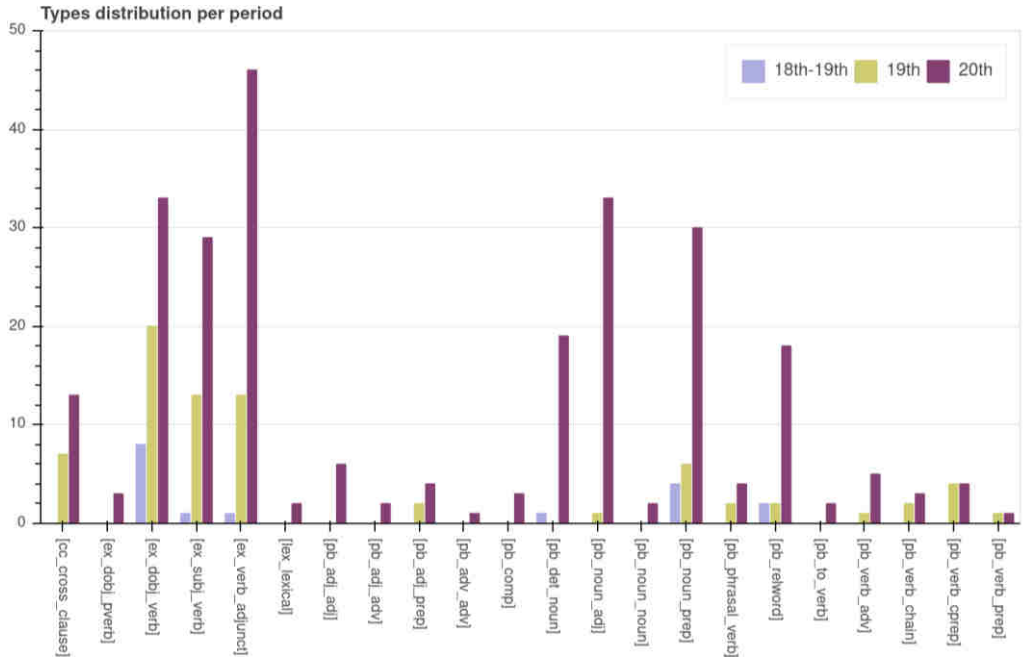

Figure 10: Distribution of each enjambment type by century in the reference corpus (number of occurrences)

The interest in introducing poems in other forms than sonnets and from the 18th to the 21st century lies in the overview it gives on how enjambment, language and poetic traditions evolved together.[28] The following diagram shows the distribution of types throughout centuries. Note that the 18th-19th category includes poems for which the publication date was unclear (the poet lived a turn of century (17th-18th or 18th-19th) and that the number of poems per century is unbalanced.

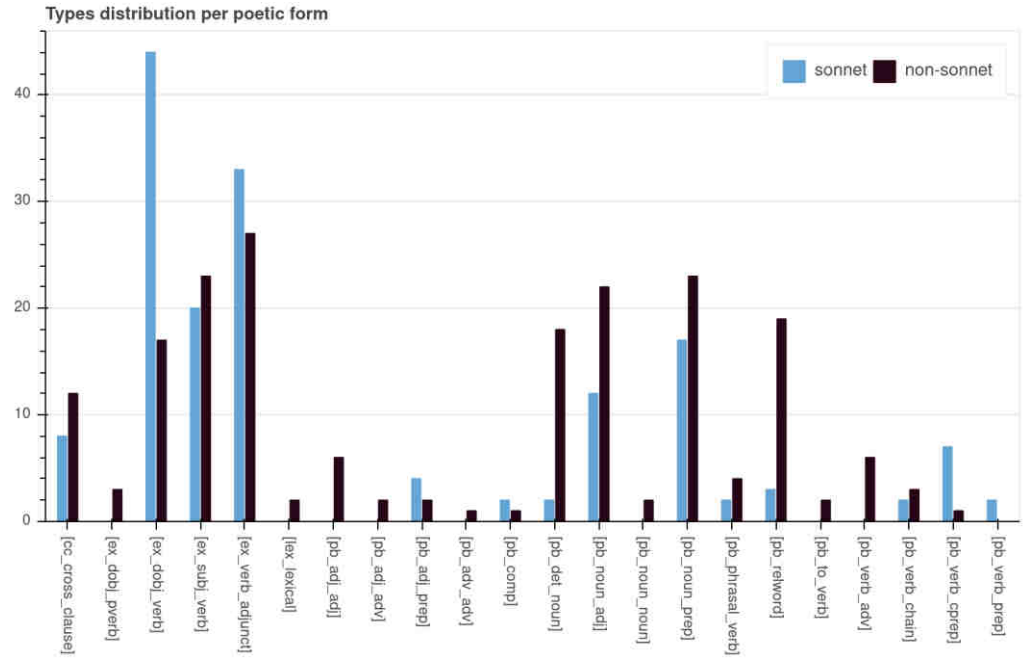

Figure 11: Distribution of each enjambment type by poetic form in the reference corpus (number of occurrences)

[28] Although the corpus is made of poems from the 14th to the 21st century, only poems from the 18th to the 20th are used in the test corpus for two reasons: First, 14th to 17th century poems tend to present old spelling or outdated syntactic structures, which would be unmanageable for tools like spaCy, typically trained on more modern English. Second, there are fewer canonised poets from the 21st. Furthermore, poems from the 21st rarely belong to the public domain. Thus, 21st poems are not included in the test corpus either.

5.2. Annotations

Each selected poem was numbered, and an annotation template was inserted at the end of the file in order to make the annotation process as seamless as possible. Concretely, given the original stanza

> "Hold fast to dreams
> For if dreams die
> Life is a broken-winged bird
> That cannot fly"
>
> Langston Hughes, "Dreams"

the empty annotation file looked like

4.Hold fast to dreams

5.For if dreams die

6.Life is a broken-winged bird

7.That cannot fly

01 02 [] [] []

02 03 [] [] []

03 04 [] [] []

This annotation template takes three types of annotations: syntactic annotation, semantic (also called meaning annotations in the annotation guide) and stylistic annotation. The annotator will enter the relevant types for each line in the square brackets in the template. Examples are provided in Appendix 3.

The syntactic annotations are concerned with Quilis and Spang's types (plus those specifically added for English), and the semantic annotations correspond to Golomb's typology. The stylistic annotations are used to distinguish enjambment cases where a pause occurs in the middle of the second line from cases where the pause only occurs at the end of the second line; this distinction was drawn by Quilis (1964) among others.

Details about the annotation process and types can be found in the annotation guide (Monget, 2020) or in Appendix 2. The annotation scheme follows the same guidelines than those of Ruiz et al. (2017).

Out of the three types of annotation, only the syntactic ones are identified by the tool for now.

5.3. Redefining the types

Sections 1.3.2 and 1.3.4 described Quilis and Spang's types for Spanish, and section 4.2 pointed out that some types absent from the Spanish typology[29] may exist in English. While annotating the corpus, it became clear that this hypothesis was true and that new types needed to be taken into account to develop a tool to appropriately annotate enjambment in English poetry. However, none of Quilis and Spang's types were withdrawn from the typology. No types were added to the cross-clause and lexical categories; additions took place within the phrase-bounded types. Note that the goal here is to try and provide a tool to detect enjambment to help study its stylistic effect,

[29] Spanish typology refers to the typology used in by Ruiz et al. (2017), which is a combination of both Quilis' and Spang's types.

not redefining the phenomenon. As such, some types that are presented below may not be considered as enjambment by other scholars, and they can decide to ignore suggestions made by the tool.

5.3.1. Challenging cases

While some cases of enjambment were pretty clear because they occurred between two adjectives modifying the same noun for instance, there are some cases of run-on lines (meaning the line was not end-stopped) that would require a more in-depth analysis in order to determine whether enjambment is taking place or not.

5.3.1.1. The case of adverbial subordination

Subordinate clauses are dependent on a main clause, and as such, they cannot stand alone: they always call for completion if read first or bring more information if read last. It is not rare to come across a full subordinate clause corresponding to a whole line, where the line is not end-stopped. We find determining the enjambment status of line-pairs involving adverbial subordinates challenging, and we discuss the problem and solution adopted.

If the adverbial subordinate clause comes before the main one and corresponds to a full line, the pause seems to be natural.

> “Hold fast to dreams
>
> **For if dreams die**
>
> **Life is a broken-winged bird**
>
> That cannot fly”
>
> Langston Hughes, “Dreams”

In this example, *“if dreams die”* is the adjunct of condition of *“life is a broken-winged bird that cannot fly”*. In the course of this thesis, eight people were asked whether they would do a pause in some ambiguous cases. In this case, all of them answered that they would indeed realise a pause at the end of *“for if dreams die”.*

On the other hand, when the adverbial subordinate clause comes after the main clause, as in *“You need a gun/ to kill a mockingbird”*, it seems less natural to perform a pause between the two. Even though the enjambment is retrospective, it would seem unnatural to those asked to realise a pause between the subordinate and the main clause. Yet, depending on the intent of the reader (meaning whether the reader wants to emphasise *“You need a gun”* for example), a pause may seem natural.

Assuming no emphasis is made while reading the poem, the hypothesis made here is that when an adverbial subordinate corresponding to a full line is followed by the main clause it is subordinated to (in other words, when the subordinate is in initial position), there is indeed an expectation that the line will continue. However, this expectation is not due to an enjambment, as suggested by the fact that several speakers consulted would make a pause after the subordinate. Rather, the sense that the sentence will continue after the initial subordinate is due to the sentence not being complete until its main clause has been uttered. On the other hand, when the adverbial subordinate follows the main clause, then a pause is rarer and there is an enjambment, even if the effect is relatively weak compared to other types.

It is not possible to confirm this hypothesis as part of this thesis, however, it could be tested by interviewing native speakers like Quilis did for Spanish.

5.3.1.2. The case of coordination

Unlike subordinate clauses, coordinate clauses are independent, and as such, it should be easy to rule out enjambment in case of run-on lines, since there are no syntactic dependencies between the two clauses. Yet, there are some tricky cases as well. For instance:

> "There was a gate I had leaned at for the view
>
> And had just turned from when I first saw you"
>
> Robert Frost, "Meeting and Passing"

In this example, there is no subject nor object for *"had turned from"* in the second line,[30] but they can be inferred from the first line, thus corresponding to both an expansion between the verb and the subject and an expansion between the verb and the object. The coordination allows the writer to remove them, but the two lines could be perfectly independent if the coordination were to be removed:

> "There was a gate I had leaned at for the view.
>
> I had just turned from it when I first saw you"

If possible, such cases of coordination are to be considered as an expansion by the tool. However, other types of coordination should not be taken into account by the tool. For instance,

> "I was curled fetally
>
> and yet I held a bayonet"
>
> Anne Sexton, "Bayonet"

is not taken into account in the typology, as *"held"* has an explicit subject in the second line.

5.3.1.3. A blurred definition

A relatively large number of run-on lines with enjambment potential are left untagged because it was not clear whether they were just run-on lines or actual enjambments due to the mixed definitions of the phenomenon.

Most definitions describe enjambment as taking place between two lines, breaking apart closely related units of the sentence. Golomb talks about the occurrence of a line boundary where syntax and semantics do not allow a break. Yet, there exist instances where a syntactic relation involves elements that occur two or more lines apart. In the light of this, should, for instance,

> (A) "The living quality of
> (B) the man's mind
> (C) stands out"
>
> William Carlos Williams, "Haymaking"

still be considered as an expansion, since the subject (*"living quality"*) and the verbal phrase ("*stands out"*) occur not only in two separate lines (A, C) but that these lines are themselves separated by another one (B)? Arguably, the effect, were this case considered as an expansion, would be even weaker than a regular case of expansion involving adjacent lines.

This matter is even more relevant in the case of William Carlos Williams' "The Red Wheelbarrow":

> (A) "so much depends
> (B) upon

[30] *turn from* is considered as a phrasal verb, *a gate* would be its object.

(C) a red wheel
(D) barrow
(E) glazed with rain
(F) water
(G) beside the white
(H) chickens."

There is at least one line beside two cohesive units because one of these cohesive units is spread onto two lines: *"a red wheelbarrow"* (line C and D) is the object of *"depends upon"* (line A and B), *"glazed with rainwater"* (line E and F) is the adjectival phrase modifying *"a red wheelbarrow"* (line C and D) and so on. In such instances, can we still talk about enjambment, or is it a different phenomenon altogether?

However interesting this issue, it is not the purpose of the thesis, nor it is within its scope, to provide a new definition of enjambment: its sole purpose is to provide a tool to help study the phenomenon, and by working toward this goal, challenging issues regarding its definition are highlighted. Therefore, the tool only takes into account two lines A and B, notwithstanding potential enjambment between A and C.

One aspect of enjambment that has not been addressed in this work is how it interacts with metrics. It may be worth looking into that direction to see whether or not it would help to have a more comprehensive definition of enjambment and see how it can affect its automatic detection. Systems for automatic metrical scansion already exist and include systems developed for Spanish (Agirrezabal, 2017), French poetry (Delente and Renault, 2015) or English poetry (Agirrezabal et al., 2018).

As pointed out by Delente (2019), an important aspect of enjambment is the reader's relationship to poetry. Some people may realise a pause when some others won't, as highlighted in Hussein et al.'s study (2018), where ex-RDA and ex-RFA readers recite poetry differently. This behaviour is another factor that makes it difficult to identify certain contexts of enjambment.

5.3.2. Phrase-bounded types

Most of the new types defined for English fall in the phrase-bounded type. They are presented in the table below and discussed after:

Tag	Definition	Example
pb_phrasal_verb	Enjambment occurs between the verb and the preposition/particle related to it	It's time to pick up the broken pieces
pb_adj_adj	Enjambment occurs between two adjectives modifying the same noun	I was at that ripe old age of 45
pb_noun_noun	Enjambment occurs between the two nouns of a compound noun	Alone in my purity World without end
pb_det_noun	Enjambment occurs between a noun and the determiner related to that noun	through the trees
pb_to_verb	Enjambment occurs between a 'to' preposition and the verb it introduces	when we began our unholy alliance to test the literary waters
Tag	**Definition**	**Example**
pb_comp	Enjambment occurs between elements that are part of a comparison structure	Better to be human Than God
pb_relword	Enjambment occurs between a conjunction and the clause it introduces	You wrote me that letter and I answered
pb_adv_adv	Enjambment occurs between two adverbs related to the same verb or adjective	another crab grabs him and pulls him back down

Table 2: New phrase-bounded types

In the English typology, the contexts covered by the *pb_relword* in the Spanish typology have been split into separate types; arguably the *pb_relword* type was too broad in Spanish. The *pb_relword* label itself is now only applied when a conjunction or preposition is separated from the clause it introduces. Three new categories cover the remaining contexts: First, *pb_det_noun*, which Hollander describes as one of the hardest cuts possible, and as such, it made sense to dedicate a tag for such cases. Second, *pb_comp*, which is rather broad since the enjambment can occur anywhere inside a comparative structure (*"being human is better than/being God"*, *"being human is better/than being God"*). Finally, *pb_to_verb*, which covers breaks between a *"to"* and the verb it introduces.

It can be argued that some of the new types introduced here are only subtypes of existing types, such *pb_adj_adj* (subtype of *pb_noun_adj*) or *pb_adv_adv* (subtype of *pb_verb_adv*), which is true. However, these new types are more specific and it would seem very unnatural to break them apart while reading a preversified form (i.e. presented as running text) of the context they appear in, notwithstanding the lack of syntactic relationship between them.

The *pb_adv_adv* may be contested on some other grounds as well. Indeed, it is very common to add up the same adverb to emphasise something, as in *"It is very, very, very hard to follow what you are saying"*. Yet, it would not be orally realised in the same way as *"It is very very very hard to follow what you are saying"* because the focus on the sentence is not the same; the rhythm suggested by the lack, or removal, of commas is much faster. If poets chose not to separate adverbs with commas, then their choice should be taken into account, because commas impact the way the poem should be recited.

5.3.3. Expansion types

Expansion is the other category in which types were added. As discussed in section 4.1.2, a type was added for phrasal verbs, as a complement to the one added in the phrase-bounded category. This was done to reflect the difference between an expansion involving a verb and its object and a phrasal verb and its object. The other addition made to the expansion category is a type describing enjambment occurring between a verb and its adjunct.

Tag	**Definition**	**Example**
ex_dobj_pverb	Enjambment occurs between the verb and the preposition/particle. Includes cases where the object is between the verb and the particle/preposition	It's time to pick up the broken pieces It's time to pick the broken pieces up
ex_verb_adjunct	Enjambment occurs between a verb and the adjunct prepositional or subordinate clause	If you dissect a bird To diagram the tongue

Table 3: New expansion types

5.3.4. Discarded contexts

Some enjambments involved pronouns instead of nouns, but the tools failed to identify the pronoun in these instances, so it was decided to leave such cases in the *pb_noun_** type it falls into.

Other enjambment types were either not represented enough or too hard to disambiguate (sometimes both) to be annotated for now, yet it is worth mentioning them: one occurrence of enjambment was between an adverb and the whole proposition (*"Maybe/ it was the upper case"*) and another one was between an adverb and a nominal phrase (*"Only/ my heart is lonely"*). Besides, it may be hard to determine when the adverb modifies a nominal phrase or a whole proposition with existing tools given that it is not the most common use of adverbs.

It was also contemplated to distinguish between copular and non-copular verbs, but enjambments occurring between a copular verb and its attribute being scarce and tools not providing a satisfactory output, it was left aside for now.

5.3.5. Other remarks

As explained in 5.2, the corpus was first annotated according to three different typologies: syntactic types, semantic types and stylistic ones. What was striking is that, for some cases, it was possible to draw a parallel between the syntactic configuration of the enjambment and how it affects the semantic completeness.

For instance, in case of a phrase-bounded enjambment noun/adj, if the adjective is before the

noun it modifies, then it is always a prospective enjambment, same goes for adv/adj, and subj/verb and dobj/verb if the verb is transitive. On the other hand, if the noun occurs before the adjectival phrase it is modified by, then it is systematically retrospective (see Table 4 below).

Prospective		**Retrospective**	
adj/noun	Your tiny/hands	noun/adj	A field/frozen with snow
adv/adj	Your very/tiny hands	adj/adv	Neither being much/known
adv/verb	I never/heard back	verb/adv	Move forward/anyhow

Prospective		**Retrospective**	
subj/verb	The wild flies/dance	noun/prep	The rest/from which the fur began
det/noun	through the/threes	verb/obj	Nectar fills/the bowl

Table 4: Systematically prospective or retrospective types

As mentioned above, while some of these cases are always true, such as the adj/noun case, some are not: what happens when the verb can be both transitive and intransitive? As humans, we are able to grasp in which form a verb is used. When we reach the end of the line and begin to read the next one, if the verb is used in its intransitive form, we consider the line as being syntactically complete: the enjambment is no longer prospective in this case, and the following line becomes additional information. The adv/verb case is also not always true: for example, it is quite common to repeat adverbs to emphasise something, such as in *"We never touched, never"* or *"I will always love you, always".* In such cases, we do not really expect the sentence to go on in the following line, but it can: *"We never touched, never/met"* ("An Almost Made Up Poem", Charles Bukowski). In the light of these variations and given the lack of syntactic cues to detect them, these annotations are left out of the tool for now.

6. JaDe - enJambment Detection tool: system description

JaDe, which stands for en**Ja**mbment **De**tection, is the system developed in the course of the thesis. It is meant to not only detect occurrences of enjambment but also classify them according to the Quilis and Spang's typologies, redefined to suit English specificities. The system is publicly available on github, at https://github.com/MongetE/JaDe.

The system is divided into a main, a processing and an evaluation module. The processing module handles the detection and the classification and then writes the type at the end of the first line of line pair; the evaluation one calls upon the processing one to annotate test data and outputs an overall evaluation. The main module is the command line interface of the tool. It allows users to analyse poems one by one or in batch. An overview of the system workflow is in Figure 12.

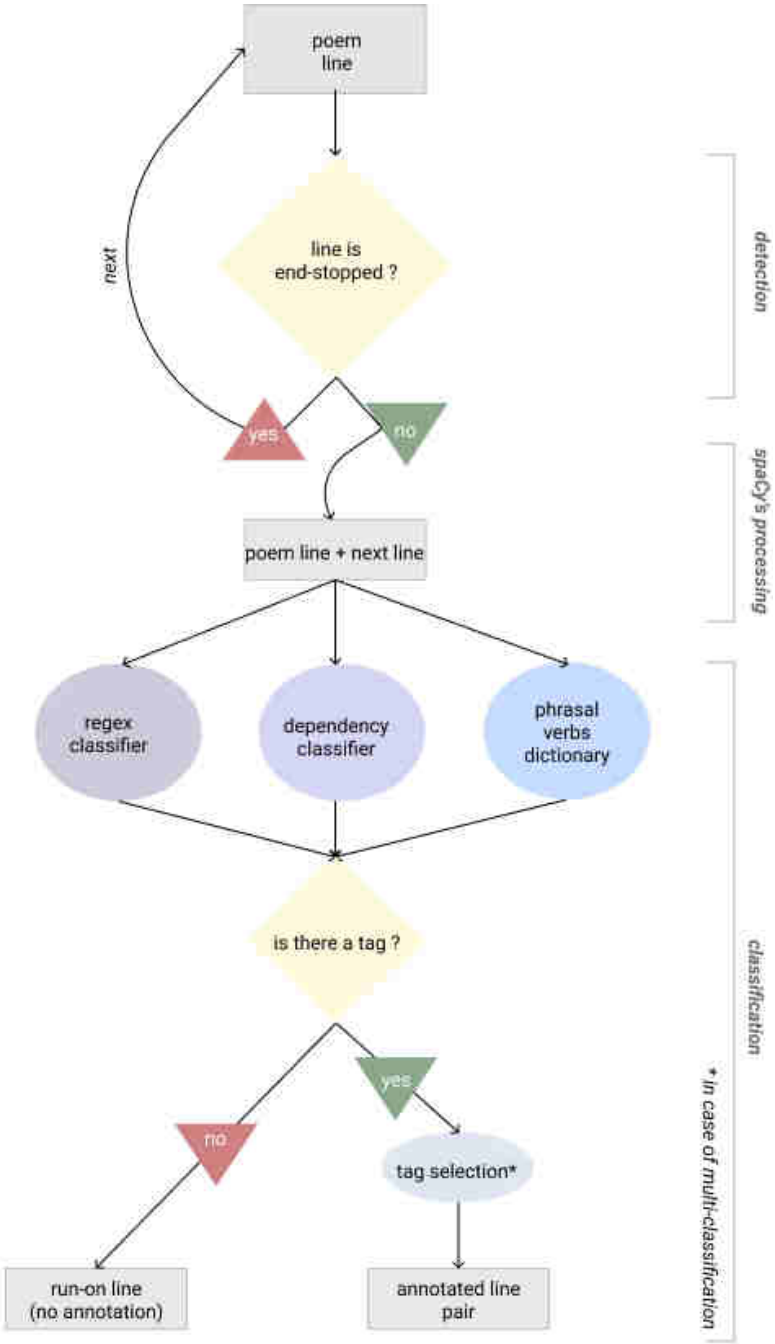

Figure 12: processing module workflow

6.1. Enjambment detection

Detecting whether a given line pair can possibly bear an enjambment is a pretty simple task. To do so, the tool iterates over each line and checks whether or not a punctuation mark (meaning a punctuation cue that indicates the end or a pause in a sentence) is found at the end of the line. If none is found, it goes ahead and tries to classify the enjambment, otherwise it is seen as an end-stopped line and it is ignored.

However, as explained in section 4.1, the lack of punctuation cues at the end of a line is not always tantamount to the presence of an enjambment. As such, the detection is not complete until the classification happens: if the classification output is not null, then the system considers the line pair as an occurrence of enjambment.

6.2. Enjambment classification

Classifying is a trickier task. The classification is done through a set of regular expressions (regex) and dependency rules, applied to the various linguistic cues provided by spaCy; the regex rules apply mainly to part-of-speech (POS) sequences. A relatively small dictionary of phrasal verbs is also included to deal with either the *pb_phrasal_verb* or the expansions involving phrasal verbs.

First, if no punctuation mark is found at the end of the line, the line pair (current line + following line) is built and then passed into the spaCy pipeline.

To pass information to the spaCy pipeline, two approaches were tested. In the first approach, the sentence in which the line pair appears was identified and passed to the pipeline. In the second approach, the entire line-pair was passed. The hypothesis was that if the model had complete sentences rather than line-pairs (which may contain complete or incomplete sentences), its annotation would be more accurate and so would the classification. However, this hypothesis was not confirmed, as the first approach performed slightly worse.

Furthermore, poetry lines tend to begin with an upper case, which can affect the spaCy pipeline. Lowercasing text sequences is often part of the natural language processing workflow and the system appears to perform better when the line pair is lowercased before processing.

Be it for the regex-based rules or the dependency rules, a special marker was necessary to signal the line break. The best option seemed to use a token among those whose POS is SPACE, since they carry close to no linguistic interest. Initially, the newline character present in the poems was kept as the line-break marker, but it appeared that it was confusing the POS tagger, so that the following word was incorrectly classified. This was not, or at least at a less frequent rate, the case with the tabulation character, although its POS-tag is also *SPACE*. Tests were done with other non-alphanumeric characters such as '%' or '\', in order to see if they would be suitable candidates as well, but they were confusing the tagger in that they were considered as punctuation, foreign words or even sometimes adjectives. Accordingly, the tab character (**\t***)* was used to represent a line boundary.

Figure 13 sums up the processing done before the classification task.

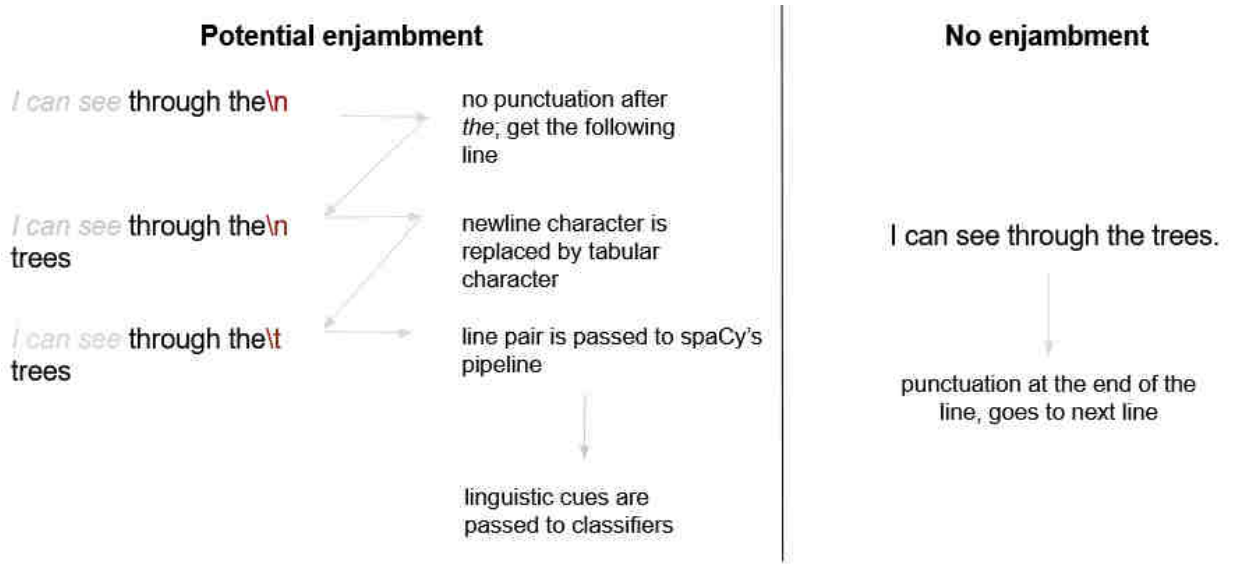

Figure 13: Preprocessing workflow before classification

6.2.1. Regular expression patterns based on part-of-speech tags

The most straightforward way to classify enjambment is using part of speech tag sequences, and regular expressions are the most efficient way to detect specific patterns. SpaCy provides both "coarse" (Universal Dependencies tagset[31]) and "fine-grained" POS tags (the Penn Treebank tagset[32]). Whereas for most enjambment types, coarse-grained tags are sufficient to handle the classification, for some (cross-clause for example), fine-grained tags are required for the classification to be more accurate.

Also note that the system only outputs one enjambment type per line-pair, for reasons that will be discussed in the following section. Bearing this in mind, and since some types are subtypes of others (*adj_noun* might be seen as a subtype of *det_noun* for instance), only the last word of the first line and the first word of the second line are taken into account by the patterns: a regex allowing an optional word before or after the enjambment might override a more specific regex. To avoid this, the patterns are reduced to the minimum. A proper enjambment-strength ranking scale to give priority to one type over another would be another way to select one single enjambment to output per line. However, this was not fully implemented as it would require carrying out enjambment perception experiments with native speakers, as Quilis did, and this was beyond the scope of this thesis.

The spaCy pipeline outputs a spaCy *Document* object, from which JaDe extracts each token along with their linguistic annotations. As a consequence, the first step before performing the regex search is to rewrite the line pair, not with the actual tokens but expressed as a sequence of POS tags. As mentioned in the previous paragraph, both coarse-grained and fine-grained tags are used in the detection patterns, so both types of tags are used when rewriting the line-pair. For instance, the *pb_det_noun* type case is handled thanks to the following pattern: DET SPACE NOUN. The following example, "through the \t trees", would give POS-sequence "ADP DET SPACE NOUN" and so the regex search would return True and the *"pb_det_noun"* would be assigned to the line pair (Fig. 14).

[31] https://universaldependencies.org/u/pos/index.html
[32] https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

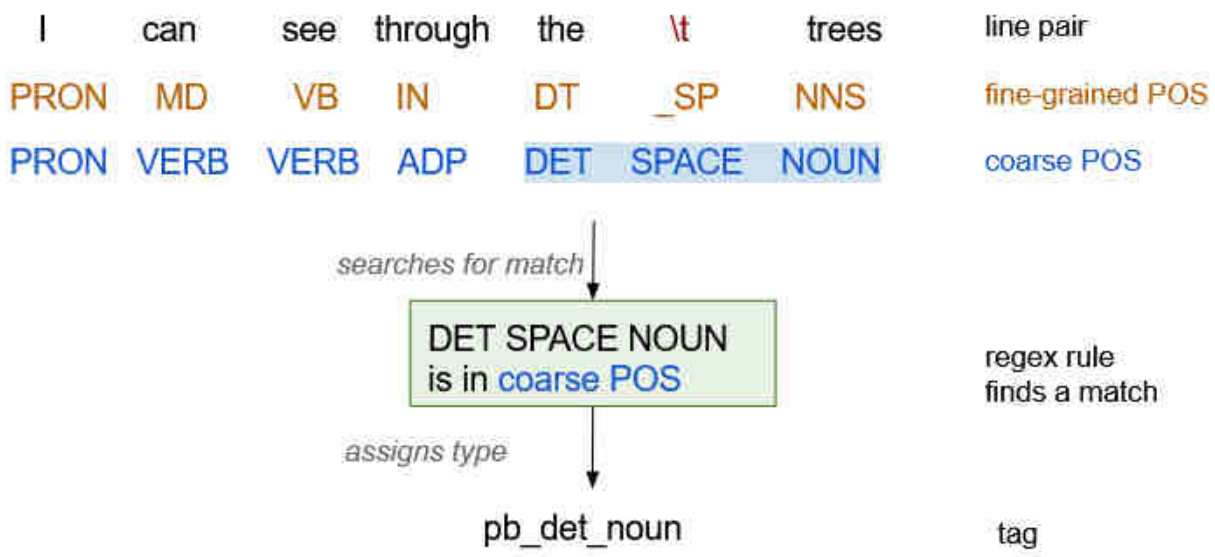

Figure 14: Application of a POS-based regex pattern

The regular expression patterns can deal with sixteen types (out of the 22 proposed in the typology (76% of types)). See table 5 below.

Types	**Pattern**
pb_det_noun	DET SPACE NOUN
pb_noun_noun	NOUN SPACE NOUN
pb_adj_noun	ADJ SPACE NOUN
pb_noun_prep	NOUN SPACE ADP
pb_adj_adv	RB.? _SP (JJ.?|VBN)
pb_to_verb	TO _SP VB.?
pb_comp	(RB|JJ)[RS](\w*){0,3} _SP(\w*){0,4}((JJ|RB)[RS]|IN)?
cc_cross_clause	NN _SP (WDT|WBG)
pb_v_chain	VERB SPACE AUX | AUX SPACE VERB
pb_adv_adv	ADV SPACE ADV
pb_verb_adv	VERB SPACE ADV | ADV SPACE VERB
pb_verb_cprep	VB.? _SP TO
pb_verb_prep	VB.? _SP IN
ex_subj_verb	NN(.+)? _SP VB[^NG].?
ex_dobj_verb	VB.? _SP (\w+){0,2}NN(.+)?

Table 5: Types detected by regex

In spite of the reservations expressed in 4.1.1, some expansion types are included in the POS-based classifier: it is assumed that a noun immediately followed by a verb is the subject of that

verb or a noun following two optional words after a verb is the object of that verb and that in case of otherwise, they are sufficiently scarce to not have a significant impact on the tool's performances.

6.2.2. Dependency rules

As mentioned in section 4.1.1.1, POS may not be enough to classify every type of enjambment, especially the expansion ones. Therefore, rules exploiting the dependencies were implemented as well.

These rules combine dependencies and POS in some cases, to ensure that the classification is as thorough as it possibly can. Table 6 gives an overview of which cues are used for each rule/enjambment type pair (dependencies or dependencies + POS).

In most rules, only the last word of the first line and the first word of the following line are taken into account, since the further a word from its head, the softer the enjambment. Besides, taking the whole line pair into account could result in a multi-classification; i.e. assigning more than one type to the line pair. The main reason for multi-classification is line length: the longer the line, the more tokens, the more dependencies. While it is not excluded to allow more than one type in the future, the reference corpus was annotated with only one type per line pair. For the purpose of providing a truthful evaluation, only one type, the alleged most striking one, is kept (cf. section 6.2.4). Thus, reducing the fragment to which the rule is applied reduces the risk of finding several types. However, for some types, such as expansions, the classifier performed significantly better if the whole line pair is taken into account. Accordingly, for detecting expansions, the whole line was taken into account.

Type	**Cues**
last word of line 1 and first of line 2	
pb_noun_noun	dependency + pos
pb_det_noun	dependency
pb_noun_adj	dependency
pb_verb_prep	dependency + pos
pb_verb_chain	dependency
pb_noun_prep	dependency + pos
pb_adj_adv	dependency + pos
pb_verb_adv	dependency + pos
whole line pair	
cc_cross_clause	dependency + pos
pb_phrasal_verb	dependency
ex_subj_verb	dependency

Type	Cues
ex_dobj_verb	dependency
other	
pb_relword	dependency + pos
ex_verb_adjunct	dependency + pos

Table 6: Types detected and cues used in the dependency classifier

For instance, considering "I can see through the \t trees": the most obvious type of enjambment is *pb_det_noun*. In order to classify the line-pair as such, the system requires the information presented in Table 7 and processes it as described in Figure 15:

tokens	I	can	see	through	the	\t	trees
token_position	1	2	3	4	**5**	**6**	**7**
dependency	nsubj	aux	ROOT	prep	**det**		pobj
has_children	False	False	True	True	False		**True**
children			I, can, through	trees			**the**
coarse_pos	PRON	VERB	VERB	ADP	DET	SPACE	NOUN

Table 7: Information required by the dependency classifier. In bold, the information used by the rule.

```
(0) line_break_position = 6

(1) for token in tokens:
(2)     if has_children is True:
(3)         for child in children:
(4)             if child_position == (line_break_position - 1) \
                    and token_position == (line_break_position + 1):
(5)                 if child_dependency in ['det', 'poss']:
(6)                     types = 'pb_det_noun'
```

(0) *\t position in the table above*
(1) *go through each word of the line pair and its associated information*
-> `[i, can, see, through, \t, the, trees]`
(2) **if the word has children...**
-> `[False, False, True, True, None, False, True]`
(3) *go through children*
-> `[i, can, through], [trees], [the]`
(4) **... and there is only one character (\t) between the head word and its child...**
-> `child_position = 5 & token_position = 7 -> [the, trees]`
(5) **... and the child's dependency is one of these: *det* or *poss***
-> `{the: det}`
(6) **assign type *pb_det_noun***

Figure 15: Pseudo-code for the *pb_det_noun* classification using dependency parsing

6.2.3. Phrasal verb dictionary

Phrasal verbs were one of the potential challenges discussed in section 4.1.1.2. The tool aims at suggesting a potential break inside a verbal group involving phrasal verbs. However, in some cases, additional information may be required ("*back up someone*" is a phrasal verb but "*from the back up to the neck*" is not): the simplified model used within the tool may have some limitations.

A list of roughly 200 phrasal verbs was gathered thanks to the EnglishClub website,[33] briefly introduced in section 4.4. As pointed out, there exist many more phrasal verbs and a list of 200 is of course limited; the same site advertises a book with over 1,000 phrasal verbs in context. However, given the small number of occurrences of enjambments involving phrasal verbs, it was not a priority to gather more.

The dictionary accounts for both the *pb_phrasal_verb* and *ex_dobj_pverb* enjambment types.

The system iterates through each verb in the list and generates regex patterns. For example, for phrasal verb *fill in*, the patterns would look as follows (recall that the tab character represents the line boundary in the pre-processed text format used by the system):

- expansion pattern **a**: the object is inserted between the verb and the particle
  Example rule: fill(.?ing|e?d|t)?(\w*){1,3}\t(?\w*){0,3} in
  Example match: fill the\tform in
- expansion pattern **b**: the object comes after the particle
  Example rule: fill(.?ing|e?d|t)? in\t(?\w*){1,3}
  Example match: filled in\tthe form
- phrase-bounded: the object comes after the particle AND the enjambment breaks apart the verb and the particle
  Example rule: fill(ing|e?d|t)\tin
  Example match: fill\tin [the form]

The first two are meant to encapsulate the different configurations for the expansion between the phrasal verb and its direct object and the last one is meant to capture the breaking-apart of the verb and its particle.

While these patterns can handle the *-ing* form and regular past tense, some phrasal verbs are irregular. As such, simple past and past participle forms of these verbs are not accounted for.

6.2.4. Handling multi-classification

As mentioned previously, dependency rules often return more than one type, usually a phrase-bounded/expansion pair, in which case the expansion is excluded, at least for evaluation, or two types of expansion (usually dojb/verb and subj/verb).

When there are two expansions (dobj/verb, subj/verb) and no phrase-bounded types, priority is given to the *ex_verb_subj* over the *ex_verb_dobj*, since a sentence might seems complete if the object is missing, especially if the verb can be intransitive, suggesting that an expansion between the verb and the object could be softer than between the verb and the subject. If one of the two expansions is *ex_verb_adjunct*, priority is given to the other type of expansion.

However, in a few rare cases, several phrase-bounded types are also returned. In the future, it

[33] https://github.com/MongetE/JaDe/blob/master/JaDe/resources/phrasal_verbs.txt

should be possible to keep all of them in the final tool's output, but since our manually annotated reference set only has one type per line, only one type is added to the line pair for now. As such, given that the regex rules only return one type, this type is preferred over the dependency-inferred ones. Furthermore, the regex rules only consider the last word of the first line and the first of the second one, thus making types returned by these rules more accurate (or at least stronger) than those returned by the dependency rules, since the latter take a bigger span of the line pair into account.

Although it makes sense that a line-pair involves more than one type of enjambment, giving the choice to the user to return them all or only the most striking one should be possible using a ranking scale inspired by Hollander's spectrum (see 1.1). Nevertheless, Hollander's spectrum is not comprehensive, and creating a reliable one would require interviewing native speakers, like Quilis did, to avoid biases.

Note that it is highly possible that the tool outputs several types for the same line pair outside of those encountered during the evaluation process.

6.3. Running JaDe

The main module allows users to run the processing module. It can be used to annotate a single file or a whole directory, and comes with a few other options, such as saving the results or choosing a spaCy model for annotation. See https://github.com/MongetE/JaDe/wiki/Running-JaDe for further details.

JaDe can also be run in evaluation mode thanks to a dedicated main module. In that case, it will output evaluation metrics (precision, recall, F1), per type and averaged, comparing the system's results to the manual reference annotations, which are also given within the system's resources. The evaluation optionally outputs confusion matrices. The evaluation mode allows us to select the components that will apply (regex, dependencies, phrasal verbs or all). This is useful in order to assess each module separately, and to better guide development, when trying to improve the system and its resources.

7. Evaluation and discussion

The tool performs two distinct actions: enjambment detection and enjambment classification. The scikit-learn native function *classification_report* is used to evaluate both tasks.

7.1. Enjambment detection

As described in 6.1, enjambment detection consists in determining whether a line-pair has an enjambment or not, irrespective of its type. The results are as follows:

	Precision	Recall	F-score	Occurrences
not enjambment	0.857	0.930	0.892	598
enjambment	0.861	0.737	0.794	354
accuracy			0.858	952
macro avg	0.859	0.833	0.843	952
weighted avg	0.858	0.858	0.856	952

Table 8: Enjambment detection measures

Note that the “not enjambment” row embraces both end-stopped lines and run-on lines without enjambment.[34] Overall, results are satisfactory.

7.2. Enjambment classification

The following section presents the evaluation of different classification methods. Section 7.2.1 only includes the POS-based classifier (sometimes called regex classifier), 7.2.2 includes the results of the dependency classifier and section 7.2.3 includes the results of all three classifiers, meaning that the results combine all three classification methods. The results for the dictionary classifier are not presented in a specific section because it only involves two types, both of them being very scarce.

The results[35] were obtained with spaCy’s smallest model.

The number of enjambment occurrences (*support* column) varies from one classifier to another: for each classifier, this number amounts to the total of manual annotations whose tag is a type supported by the classifier.

[34] Section 5.3.1 discusses cases of run-on lines that do not show enjambment.

[35] Note that results for the regex-based version are also reported on github, as part of the tool’s v1.0 release (https://github.com/MongetE/JaDe/releases/tag/v1.0). The results reported in the thesis are slightly better, since they were obtained after several improvements in the source code and after replacing the newline character by the tabulation character as the line-boundary marker for poem preprocessing prior to rule application; the reason for this change of marker was described in section 6.2.

7.2.1. Regular expression classifier

The regex version of the tool, using only parts of speech, is able to detect 16 types out of 21[36] (cf. section 6.2).

	precision	recall	f1-score	support
micro avg	0.782	0.477	0.592	256
macro avg	0.776	0.546	0.578	256
weighted avg	0.883	0.477	0.602	256
cc_cross_clause	1.000	0.450	0.621	20
ex_dobj_verb	0.920	0.377	0.535	61
ex_subj_verb	0.741	0.465	0.571	43
pb_adj_adj	1.000	0.500	0.200	6
pb_adj_adv	1.000	0.667	0.667	2
pb_adv_adv	1.000	0.500	0.667	2
pb_comp	0.400	0.667	0.500	3
pb_det_noun	1.000	0.550	0.710	20
pb_noun_adj	1.000	0.512	0.583	34
pb_noun_noun	0.125	0.500	0.200	2
pb_noun_prep	0.923	0.600	0.727	40
pb_verb_adv	0.667	0.667	0.667	6
pb_verb_chain	0.800	0.800	0.800	5
pb_verb_cprep	0.667	0.250	0.364	8
pb_verp_prep	0.182	1.000	0.308	2
pb_to_verb	1.000	0.500	0.667	2

Table 9: Results for the POS-based classifier (averaged and per-type precision, recall, F1)

Overall, most types yield satisfactory results: with the exception of the *adv_adv*, *verb_prep*, *verb_cprep,* and *noun_noun*, they all have an F1-score greater than 0.5.

[36] There are actually 22 types in the annotation guidelines, but the lexical type is not dealt with by the classifiers in the current implementation.

The *pb_verb_cprep* is close to the *pb_verb_prep* one: in both types, a verb and the preposition modifying that verb are involved. The former implies a lexical bound, somehow similar to that of a phrasal verb and its particle, and the latter involves a ditransitive verb. Most prepositions can be used in both cases, thus, using patterns to differentiate between them is tricky. However, "*to*" (as in *expect to*, *listen to*…) has its own POS in our tagset, unlike other lexically-related prepositions such as "*at*" (*look at, yell at*), "*on*" (*groove on*, *crunch on)* or "*for*" (*break for*, *argue for*). Therefore, verbs requiring "*to*" as part of their lexical scheme are the only ones accounted for by the system, which can explain the low performances of the *pb_verb_cprep* type.

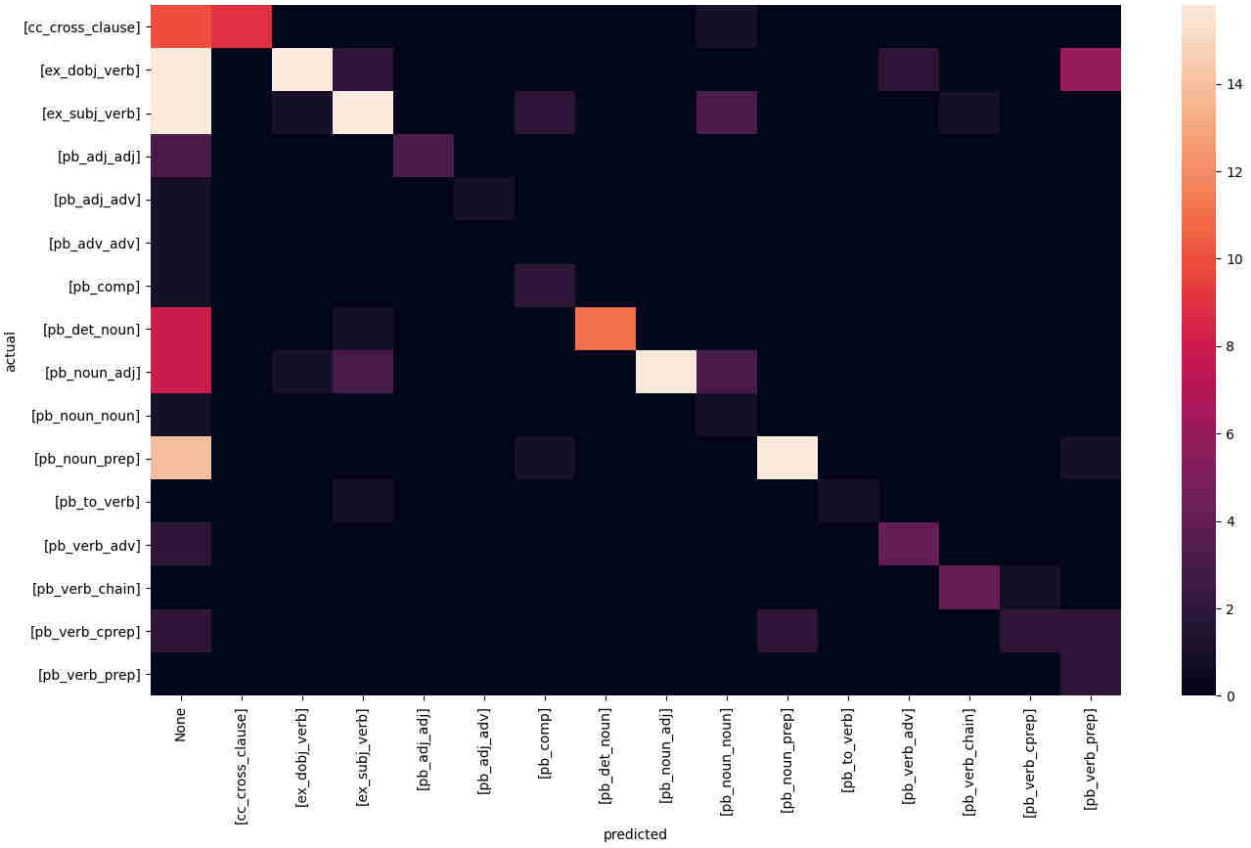

Figure 16: Confusion matrix for the POS-based classifier

According to the confusion matrix (Fig. 16), the *pb_verb_prep* is also often mistaken for *ex_dobj_verb*. This is probably because of the regex pattern for the *ex_dobj_verb*: it allows up to two words before the noun (since an object can be a nominal phrase, so there might be a determiner and an adjective before the actual object). Besides, part of the POS sequence given by a verb followed by a prepositional phrase (PREP DET NOUN) is similar in length to the POS sequence found in a nominal phrase (DET ADJ NOUN).

The expansion types are particularly well detected bearing in mind the reservations expressed throughout section 4. Nonetheless, for more than half the cases of each expansion type, the tool remains silent (cases where the expected type is *ex_*_verb*, but the type actually obtained is *None* in Fig. 16).

7.2.2. Dependency classifier

Dependency relations are the other main information that can be used to classify enjambment. Given the nature of enjambment, that is a break between related units of a sentence, using dependencies could be expected to allow for a more accurate representation than using POS, since adjacent POS can be syntactically unrelated (see example in section 4.1.1.1). However, the

dependency classifier achieves low results:

	precision	recall	F1-score	support
micro avg	0.618	0.374	0.466	329
macro avg	0.649	0.333	0.390	329
weighted avg	0.712	0.374	0.456	329
cc_cross_clause	0.667 (-0.333)	0.500 (+0.050)	0.571 (+0.050)	20
ex_dobj_verb	0.806 (-0.114)	0.475 (+0.098)	0.598 (+0.063)	61
ex_subj_verb	0.521 (-0.220)	0.581 (+0.116)	0.546 (-0.022)	43
ex_verb_adjunct	0.667	0.033	0.063	60
pb_adj_adv	1.000	0.500	0.667	2
pb_adj_prep	1.000	0.167	0.286	6
pb_det_noun	1.000	0.400 (-0.150)	0.552 (-0.188)	20
pb_noun_adj	1.000	0.500 (-0.059)	0.667 (-0.050)	34
pb_noun_noun	0.000 (-0.125)	0.000 (-0.500)	0.000 (-0.200)	2
pb_noun_prep	0.708 (-0.215)	0.425 (-0.175)	0.531 (-0.196)	40
pb_phrasal_verb	1.000	0.167	0.286	6
pb_relword	0.667	0.364	0.471	22
pb_verb_adv	0.333 (-0.334)	0.167 (-0.500)	0.222 (-0.445)	6
pb_verb_chain	0.500 (-0.300)	0.200 (-0.600)	0.286 (-0.514)	5
pb_verp_prep	0.042 (-0.140)	0.500 (-0.500)	0.077 (-0.231)	2

Table 10: Results for the dependency classifier (averaged and per-type precision, recall, F1). Values in parentheses refer to the score differences compared to the POS-based classifier

As expected from the overall measures, the dependency classifier is less performant than the regex one: for most of the types also supported by the POS-based classifier (in bold), measures drop rather significantly.

There is a strong confusion between the *ex_dobj_verb* and the *ex_subj_verb* (Fig. 17). This may be due to the dependency classifier's tendency to output several types and that the choice was made to opt for the *ex_subj_verb* when both types of expansion were found.

Two types are particularly challenging for the dependency classifier: the *ex_verb_adjunct* and the *pb_verb_prep*. We explore possible reasons for this difficulty.

These two types are very similar: they involve verbs and function words. The main (if not only) difference between them is the function played by the prepositional group. In "*These are the letters*

which Endymion wrote to one he loved", "*to one he loved*" is the indirect object of the verb *write* and it cannot be moved at the begin at the sentence "**To one he loved these are the letters which Endymion wrote*". However, in "*I like to swim on Sundays*", "*on Sundays*" can be moved anywhere in the sentence (*On Sundays, I like to swim*; *I like, on Sundays*, *to swim*): it is an adjunct of time. Therefore, the first example has the potential to be a *pb_verb_prep* while the latter has the potential to be an *ex_verb_adjunct*. The difference is very subtle and the two overlap quite often (see Fig. 17): a considerable number of *ex_verb_adjunct* are tagged as *pb_verb_prep*.

Enjambments involving adjuncts could be better dealt with thanks to semantic role labelling as it gives the role of a given phrase in the sentence: the technology can give information about *who* did *what*, *when* and *how* the action was done. In other words, it can ascertain what chunk acts as the subject, object or adjunct (and its kind (location, manner, time…)) in a given sentence. AllenNLP[37] provides such a service.

Enjambments involving adjuncts could also be removed altogether as it is debatable whether they are enjambments, given their adverbial nature. When removed, the weighted average f1-score reaches 0.524 (previously 0.456), yet the measures for *pb_verb_prep* are unchanged: there are only two types affected, namely *ex_subj_verb* (slight decrease in precision and F1-score) and *pb_relword* (all measures increase). On the other hand, when the *pb_verb_prep* is removed, the measures do not increase significantly nor does it affect the *ex_verb_adjunct* measures. Therefore, the similarity between *ex_verb_adjunct* and *pb_verb_prep* does not seem to be the main reason behind the low performance of the classifier.

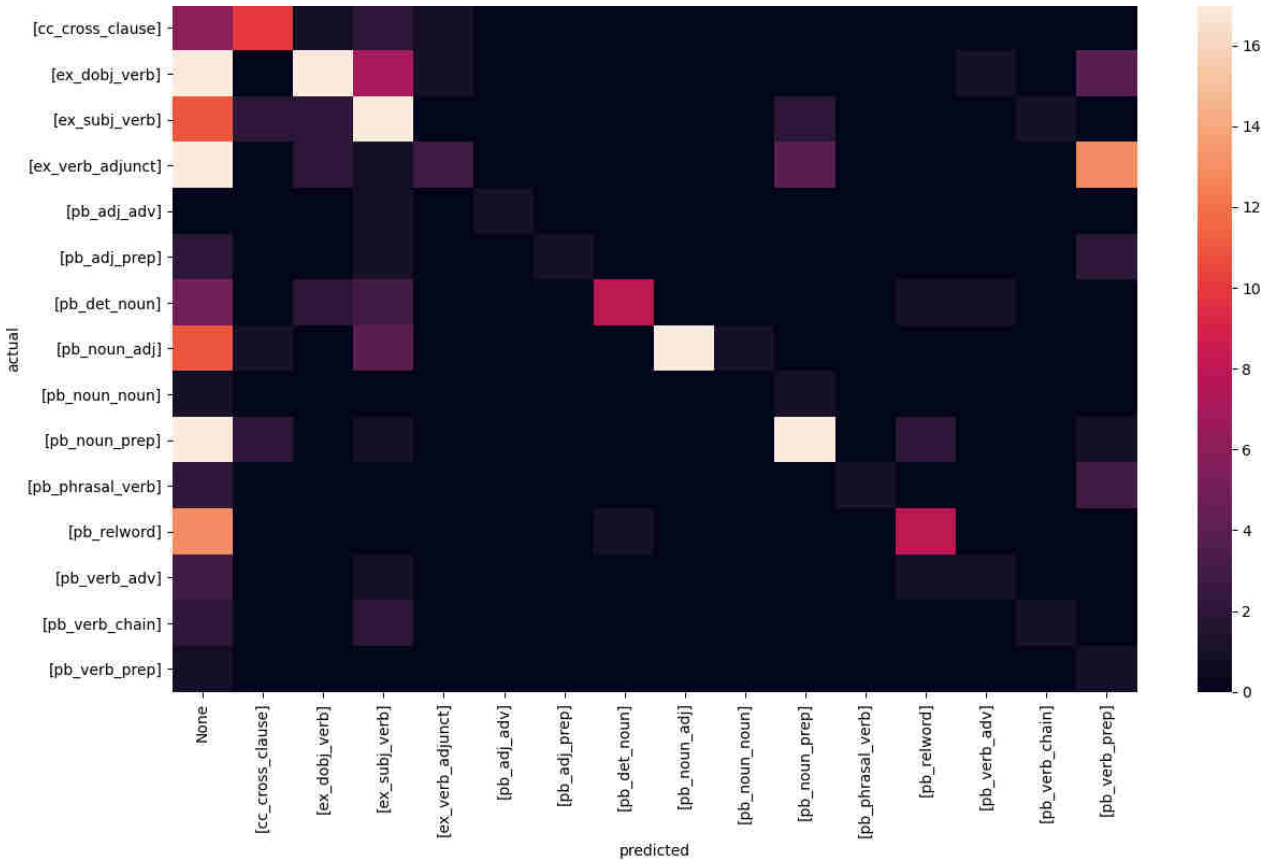

Figure 17: Confusion matrix (dependency classifier)

Another reason for these low results could be the nature of dependency parsing itself. Indeed, dependency parsing is a more complex task than POS tagging: according to the models' page,[38]

[37] https://demo.allennlp.org/semantic-role-labeling/

[38] https://spacy.io/models/en

spaCy's smaller model achieves an accuracy of 97% for POS tagging to 91.7% for dependency parsing (considering unlabelled attachment score (UAS), where the dependency arc between two units is correctly assigned but may be incorrectly labelled). However, according to Choi et al. (2015), spaCy's dependency parser accuracy drops to 87.92% when considering labelled attachment score (LAS; when the dependency arc is correctly assigned and labelled) and 89.76% according to the website. The study also shows that it performed slightly better when the punctuation was removed (+1.03%).

Despite the alleged better performances of larger spaCy models (accuracy +0.2% on both POS and UAS parsing for the largest model), when they were used to run the evaluation, the dependency classifier results did not improve and, on the contrary, decreased. While punctuation plays an important role in enjambment detection, once the line pair is passed to the classifier, the punctuation could be removed to see if performance improves.

An overall reason for spaCy's performance on this corpus can of course be that spaCy is a generic NLP pipeline, intended primarily for non-poetic texts. The non-standard features of poetry (text split into lines), uncommon lexical and syntactic usage, can be challenging for a generic NLP tool.

SpaCy models are trained on OntoNotes (Weischedel et al., 2011), a corpus mainly made of broadcast conversations and news, as well as telephone conversations, whose textual nature greatly differs from that of poems, not only in terms of lexicon but also in style. This difference in nature does not seem to strongly affect POS tagging, but it is likely troublesome when it comes to dependency parsing: an author switching whole units around for stylistic effect or the line break character (required for the classification) could mislead the parser (the newline character was an actual issue for POS tagging (see section 6.2)).

A low LAS could explain why so many occurrences of most types are left untagged by the tool (*None* in the confusion matrices): even if there is indeed a relation between a verb and a noun for example, is it an *nsubj*, *nsubjpass*, *agent*, *dobj*…?

These low results could also result from a deficiency on the rules' part: in spite of all the care put into the work, it is possible that some configurations were not taken into account, or that the way multi-classification is dealt with impairs the ability of the classifier.

A more in-depth study of the classifier's errors would help determine more clearly to what extent the error source is the system itself or spaCy's dependency parsing results.

7.2.3. Combined classifiers

On their own, none of the classifiers is able to deal with the 21 types: in order to deal with them all, they have to be combined. Table 11 shows the results when combining classifiers. Classifiers are combined according to a priority scale, from the most specific (dictionary-based) to the less specific (dependency-based).

	precision	recall	f1-score	support
micro avg	0.634	0.462	0.534	351
macro avg	0.721	0.501	0.510	351
weighted avg	0.738	0.462	0.522	351

tag	classifier	precision	recall	f1-score	support
cc_cross_clause	r-dep	1.000	0.600 (+0.150)	0.750 (+0.129)	20
ex_dobj_pverb	dict	1.000	0.333	0.500	3
ex_dobj_verb	r-dep	0.829 (-0.091)	0.557 (+0.220)	0.667 (+0.127)	61
ex_subj_verb	r-dep	0.588 (-0.153)	0.698 (+0.186)	0.638 (+0.067)	43
ex_verb_adjunct	dep	0.500 (-0.167)	0.017 (-0.016)	0.032 (-0.031)	60
pb_adj_adj	regex	1.000	0.500	0.667	6
pb_adj_adv	r-dep	1.000 (-0.500)	0.500 (+0.500)	0.667	2
pb_adj_prep	dep	1.000	0.167	0.286	6
pb_adv_adv	regex	0.500 (-0.500)	0.500	0.500 (-0.167)	2
pb_comp	regex	0.333	0.667	0.444	3
pb_det_noun	r-dep	0.923 (-0.077)	0.600 (+0.050)	0.727 (+0.017)	20
pb_noun_adj	r-dep	1.000	0.559	0.717	34
pb_noun_noun	r-dep	0.125	0.500	0.200	2
pb_noun_prep	r-dep	0.676 (-0.247)	0.625 (+0.025)	0.649 (-0.078)	40
pb_phrasal_verb	dep-dict	1.000	0.167	0.286	6
pb_relword	dep	0.800 (+0.133)	0.364	0.500 (+0.029)	22
pb_verb_adv	r-dep	0.571 (-0.091)	0.667	0.615 (-0.052)	6
pb_verb_chain	r-dep	0.800	0.800	0.800	5
pb_verb_cprep	regex	0.500	0.250	0.333	8
pb_verp_prep	r-dep	0.061 (-0.121)	1.000	0.114 (-0.194)	2
pb_to_verb	regex	1.000	0.500	0.667	2

Table 11: Results for all three classifiers combined (averaged and per-type precision, recall, F1). *r-dep* is used to indicate that a type is supported by both the POS-based classifier and the dependency classifier, *dep-dict* to indicate that it is supported by the dictionary classifier and the dependency classifier. Values in parentheses indicate the gain or loss of the combined regex-dependencies classifier compared to the regex-only classifier.

When combining the classifiers, the precision for cross-classifier types almost always lowers while the recall tends to go up. However, the loss is often greater than the gain, or the gain is infinitesimal. Only two types really benefit from the combination of the dependency rules and regular expression patterns, namely *cc_cross_clause* (f1-score +0.129) and *ex_dobj_verb* (f1-score +0.127).

We can also observe a slight improvement for the *pb_relword* category, which is probably the result of the regex types taking priority over the dependencies types. This behavior also explains why the measures of most of the r-dep types remain the same as in the regex classifier in the combined classifier when their measures were so low with the dependency classifier.

For ease of reading, the scale used below is 0-100. The results are mixed: some types are quite reliable (70% < f1-score < 80%) while some others cannot be trusted (0 < f1-score < 50%), and most types are in-between (cf Table 12).

Reliable (f1-score > 70%)	**Mid-reliability (f1-score ≥ 50%)**	**Unreliable (f1-score < 50%)**
cc_cross_clause	ex_dobj_pverb	ex_verb_adjunct
pb_det_noun	ex_dobj_verb	pb_adj_prep
pb_noun_adj	ex_subj_verb	pb_comp
pb_verb_chain	pb_adj_adj	pb_noun_noun
	pb_adj_adv	pb_phrasal_verb
	pb_adv_adv	pb_verb_cprep
	pb_noun_prep	pb_verb_prep
	pb_relword	
	pb_verb_adv	
	pb_to_verb	

Table 12: Enjambment types according to their reliability in F1 terms

However poor the results of some types, the system in itself still performs relatively well as a whole. As for the detection task, which roughly translates to the ability of the system to distinguish between enjambed and non-enjambed lines regardless of the type of enjambment, the system reaches an accuracy of 85.8%.

The figure below sums up the results of each classifier (only the overall f1-score is taken into account). It shows that the POS-based classifier outperforms the other two by a rather wide margin (+6 in average compared to the combined classifier and +15.60 in average compared to the dependency classifier; see Figure 18). Given that the POS-based classifier can deal with three-fourths of the types, among the strongest contexts of enjambment, it would be useful in the future if the tool gives the user the possibility to choose which classifier to use (combined or regex).

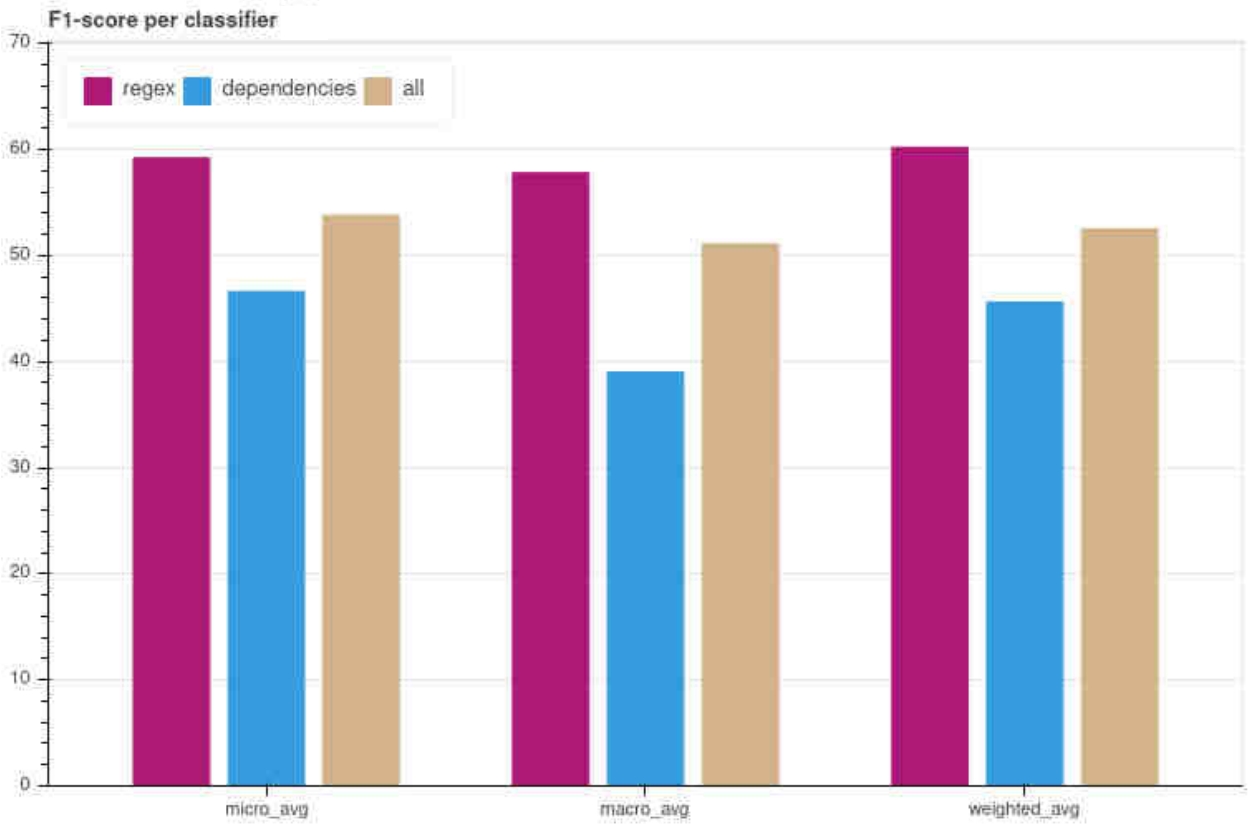

Figure 18: Overall f1-score comparison (scale 0-100)

8. Case study - using JaDe to study enjambment

This section discusses the application of JaDe to a large poem corpus. It intends to give an indication of how the tool created in this thesis can help study trends in the use of enjambment, quantitatively, based on large samples, according to variables like author, period or poetic form. The structure of the section is the following. In 8.1, we provide general remarks on the case-study corpus and some disclaimers regarding the (sometimes limited) interpretability and reliability of the tool's results. Keeping that in mind, in 8.2 we move on to a description of the distribution of enjambment types across periods in the case-study corpus, also comparing it with the manually annotated reference corpus. In 8.3 we compare the use of enjambment in individual authors from the corpus' different periods. In 8.4, we compare enjambment use across poem types, focusing on sonnets vs. non-sonnets.

Note that, besides this section, a tutorial on enjambment result visualization, which discusses additional examples, was made available on GitHub[39] and nbviewer[40] in Python notebook format.

8.1. Corpus description and remarks on the interpretability of results

A corpus of 26,501 lines by 6 different authors was built: the poems were selected randomly, within the limit of 4,400 lines per author (if the limit was reached within a poem, the poem was still added to the corpus). Doing so, the number of lines per author is homogenous, with no more than a 51-line difference.

The main criterion for author selection was the century they lived in: 2 authors per period.

author	century	nb of lines / poems	nb of run-on pairs	nb of end-stopped lines	nb of detected enjambments (combined classifiers)	ratio detected/run-on
Charles Bukowski	20th	4,447 / 102	2,782	1,222	1,612	57.93%
Oscar Wilde	19th	4,402 / 57	1,974	1,974	1,154	58.46%
Ralph Waldo Emerson	19th	4,417 / 85	907	3,061	623	68.68%
Robert Burns	18-19th	4,424 / 113	486	3,185	206	42.38%
Sylvia Plath	20th	4,407 / 137	1,497	2,116	966	64.53%
William Blake	18-19th	4,400 / 138	1,263	2,440	595	47.11%

Table 13: Description of the corpus used for the case study

The first thing we can notice is that poems from the 19th and 20th tend to have more run-on lines than earlier poems (Fig. 19), with the exception of William Blake's poems. Blake (1757-1827) is known for his extensive use of enjambments when it was frowned upon (Wolfson:68, 2003).

[39] https://github.com/MongetE/JaDe/blob/master/JaDe/jane/jane.ipynb
[40] https://nbviewer.jupyter.org/github/MongetE/JaDe/blob/master/JaDe/jane/jane.ipynb?flush_cache=true

The second interesting thing is that earlier poems seem to be challenging for the system, in the sense that the ratio detected/run-on is smaller than 50%. This ratio is smaller than what we might expect based on the ratio between enjambed lines and total run-on lines in the manually annotated reference corpus, which was ca. 75%. Notwithstanding the different scale and methodology (the reference corpus contains a large variety of authors rather than just two authors), this difference in the ratios may seem striking.

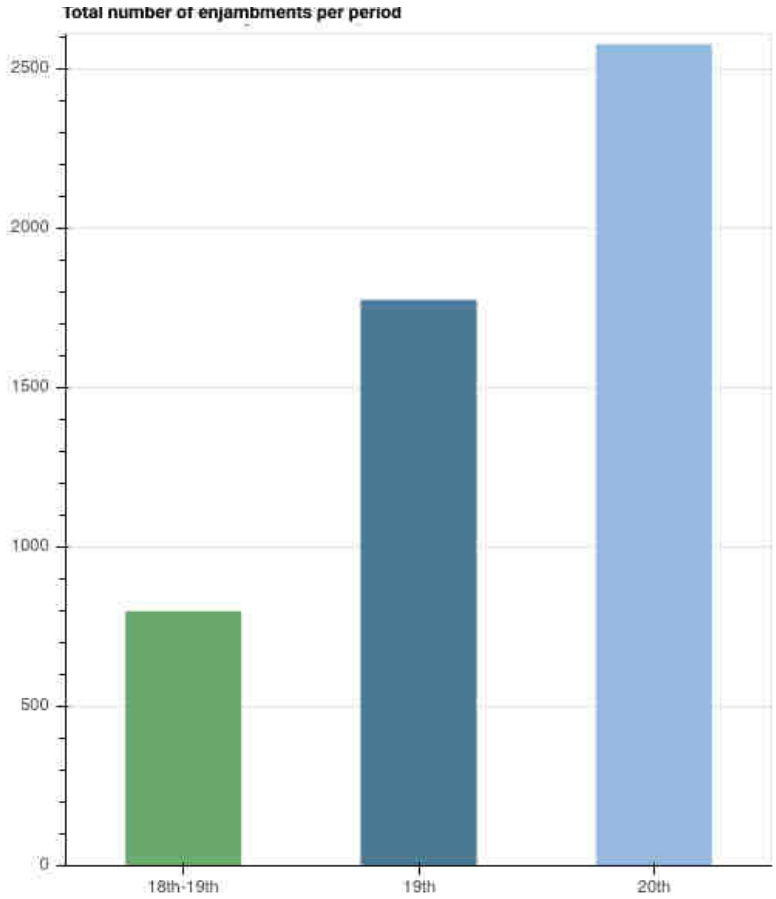

Figure 19: Total number of detected enjambments per period

Nevertheless, we saw that run-on lines are not tantamount to enjambments, and there are at least two possible reasons for lower enjambment/run-on ratios in 18-19th authors. First, enjambments were controversial (Wolfson, 2003) at least until the end of the 18th century, so run-on lines might often correspond to a full clause followed by a conjunction and another full clause. That way, there is no grammatical break from one line to another, but there is no need for a punctuation cue either: quite a lot of Burns' run-on lines occur between two clauses separated by a conjunction. What's more, as Figure 20 shows, the types detected for the 18th-19th century period are not among the stronger ones: they are mainly expansions between verbs and subjects or objects, or enjambments between a noun and the preposition modifying that noun (*pb_noun_prep* or *cc_cross_clause*). A few other types are also detected, but they are less reliable (cf. Table 12). Second, it might be due to limitations on the tool's part: as shown in section 7 (Table 8), the results of the detection are quite good, but there is still a 15% error rate. As such, some observations made in this section might be slightly biased due to the system's performance. To counter this, the analysis focuses on the most reliable types and on types with mid-reliability (see Table 12) as much as possible.

8.2. Analysis per period

The distribution of types per century in the case study corpus shown in Figures 20, 21 and 22 is

generally consistent with the distribution per century in the reference corpus. Figure 21 focuses on JaDe's most reliable types and compares their distribution in the annotated corpus and the case-study corpus.

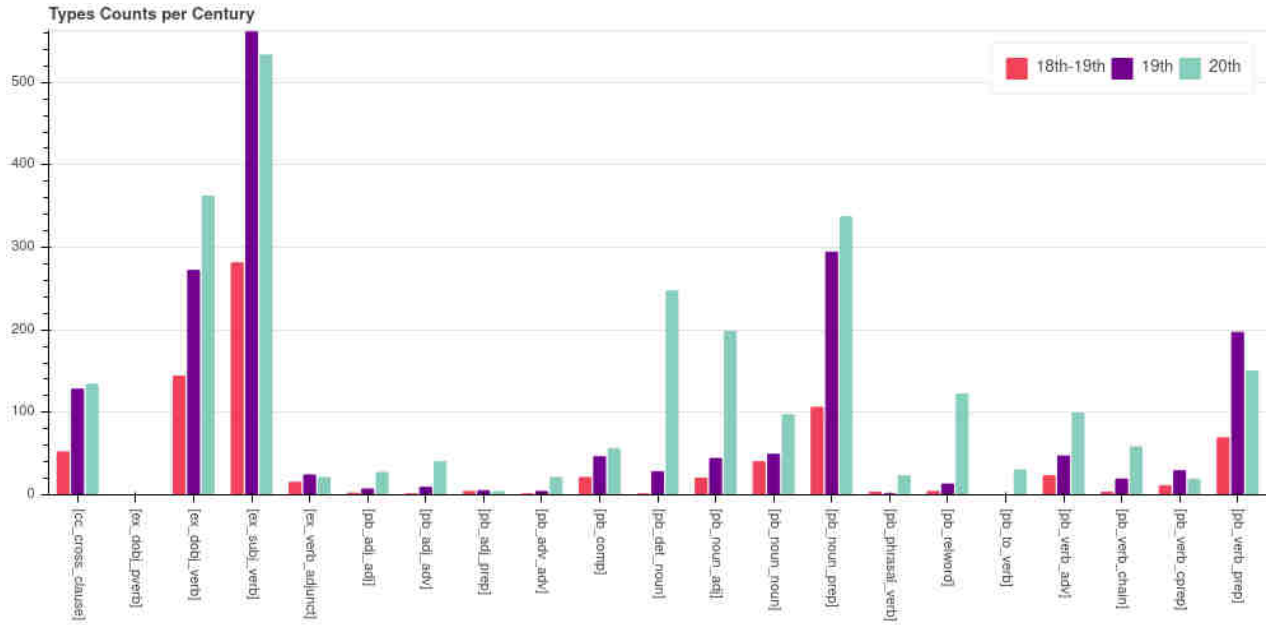

Figure 20: Distribution of types in the case study corpus

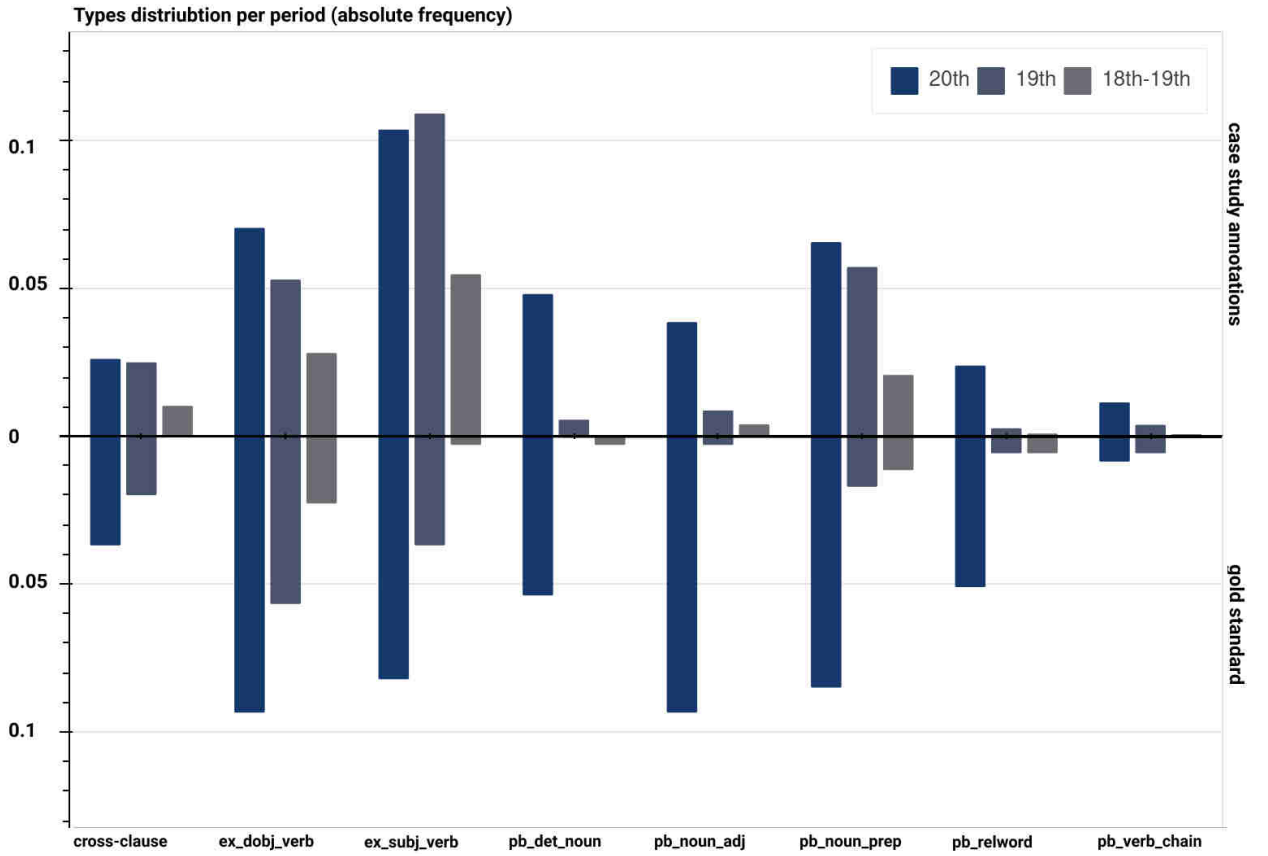

Figure 21: Distribution of a few selected types in both the case study and reference corpora. Only the most reliable types and the more common ones among the mid-reliability types are represented (according to their reference corpus frequency). Their distribution in the case-study corpus is generally consistent with their distribution in the reference corpus

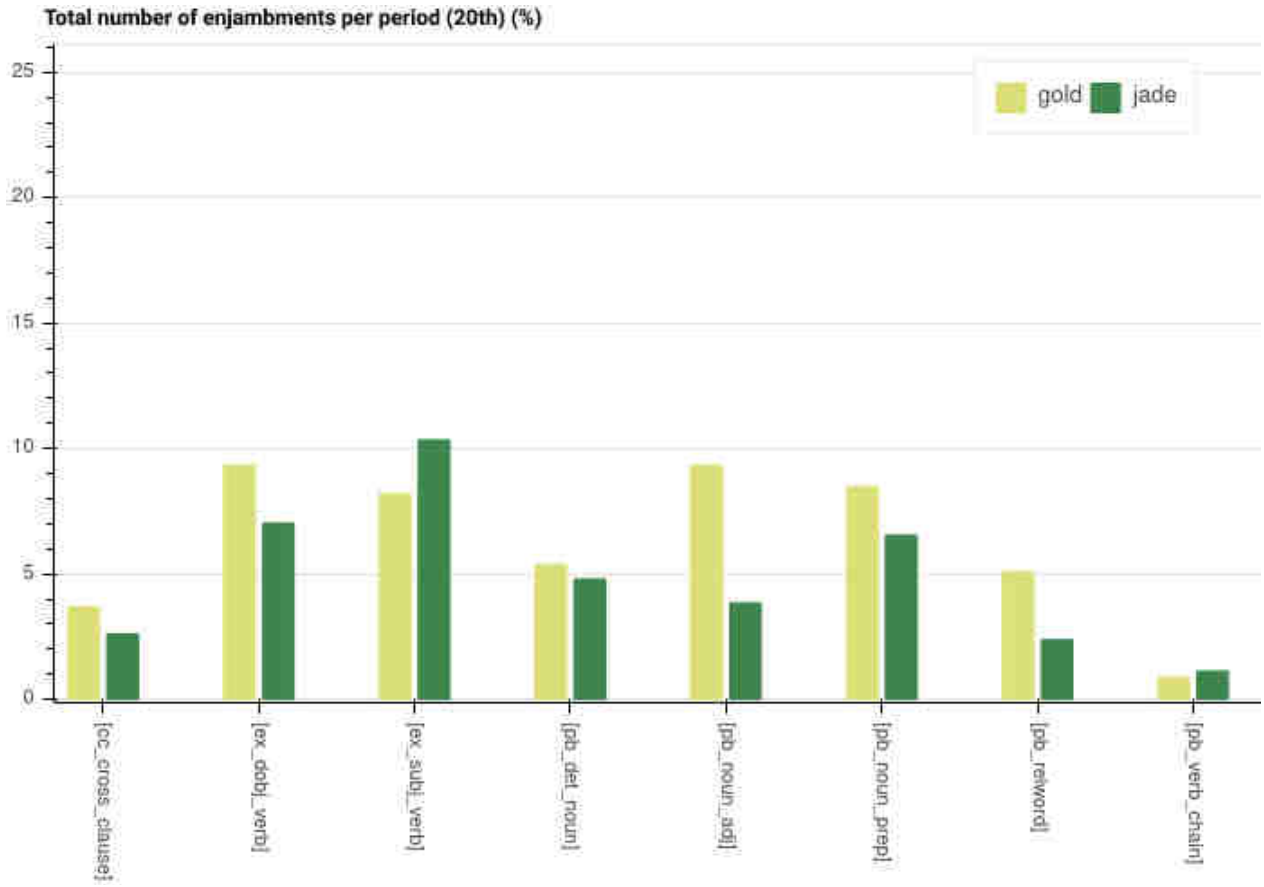

Figure 22: Distribution of a few selected types in 20th (% of each type over the number of automatically detected or manually annotated enjambment contexts). Only the most reliable and the more common types among the mid-reliability ones are represented (based on reference-corpus frequency). *Gold* refers to the annotations in the annotated corpus and *jade* to the annotations in the case-study corpus

Figure 22 focuses on the 20th century poems for the sake of example; Appendix 5 shows such details for the poems from other periods. The distribution in the case-study corpus of the most reliable and the most frequent enjambment types is similar to their distribution in the reference corpus.

Overall, although the case study and the reference corpora are still relatively small and might not be representative enough, we can see that poets from the 20th tend to use enjambments more than earlier poets and that expansions are far more common than any other types, especially in earlier poems.

8.3. Comparison between authors

Although Blake is known for his use of enjambments, it can be interesting to have a closer look at which types of enjambments he uses more, and compare his use to that of Burns', given that they lived at around the same time. As we can observe in Figure 23, Blake did not use very strong enjambments often, using mostly expansions and breaks between nouns and their complements. What is more, we can notice that Burns also did use softer enjambments, and that while Blake might indeed have used strong or unconventional[41] enjambments, such as *pb_adj_adj* or *pb_det_noun*, Burns did not. Nevertheless, *pb_adj_adj* or *pb_verb_prep* to name a few, are not

[41] Since they are the most widely spread both the reference corpus and in the case study corpus and their detection is reliable or relatively reliable, the *ex_subj_verb*, *ex_dobj_verb*, *cc_cross_clause*, *pb_noun_adj* and *pb_noun_prep* are considered here as conventional types of enjambment. *Unconventional* on the other hand refers to the less common types and the types that were not in the original typology designed for Spanish by Quilis (1964), which was the inspiration for our English typology.

as reliable as *pb_det_noun* or *pb_det_noun* (cf Table 12).

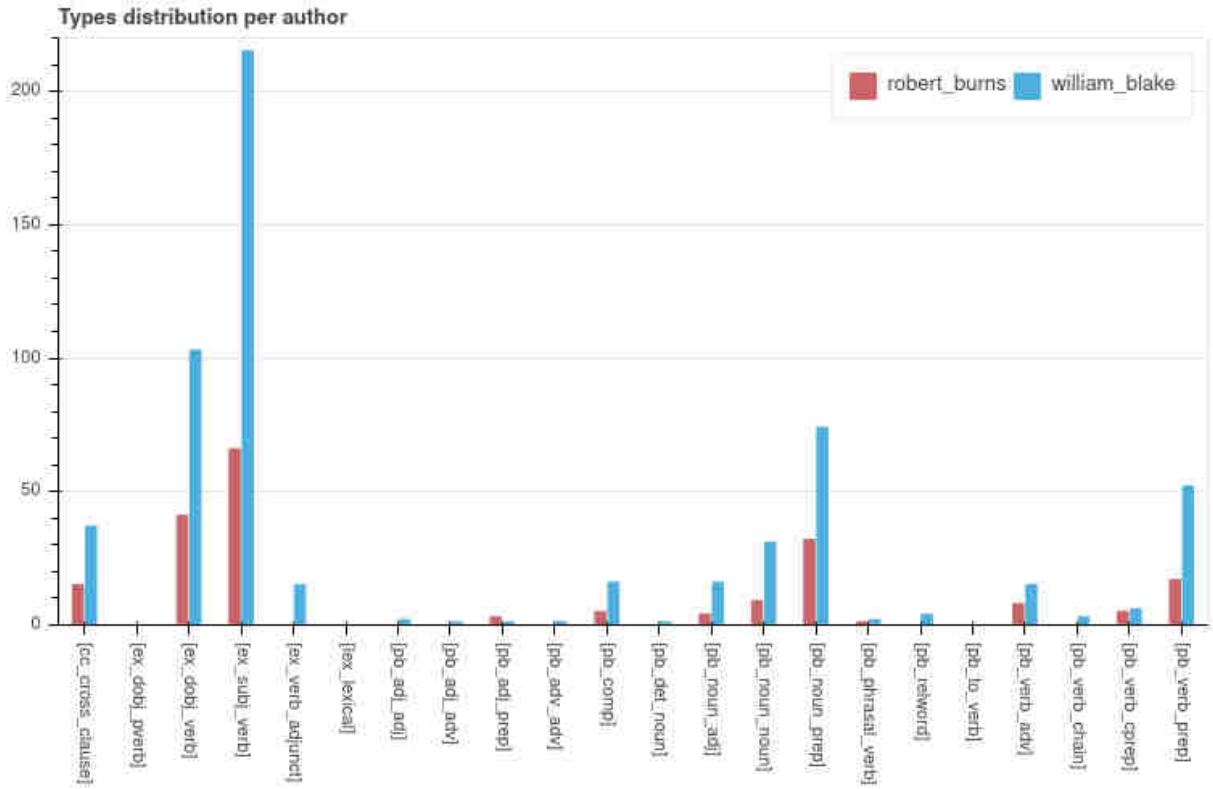

Figure 23: Distribution of types per author (Burns and Blake)

Charles Bukowski and Oscar Wilde are the other two authors who make extensive use of enjambments compared to their century peers in our case study corpus.

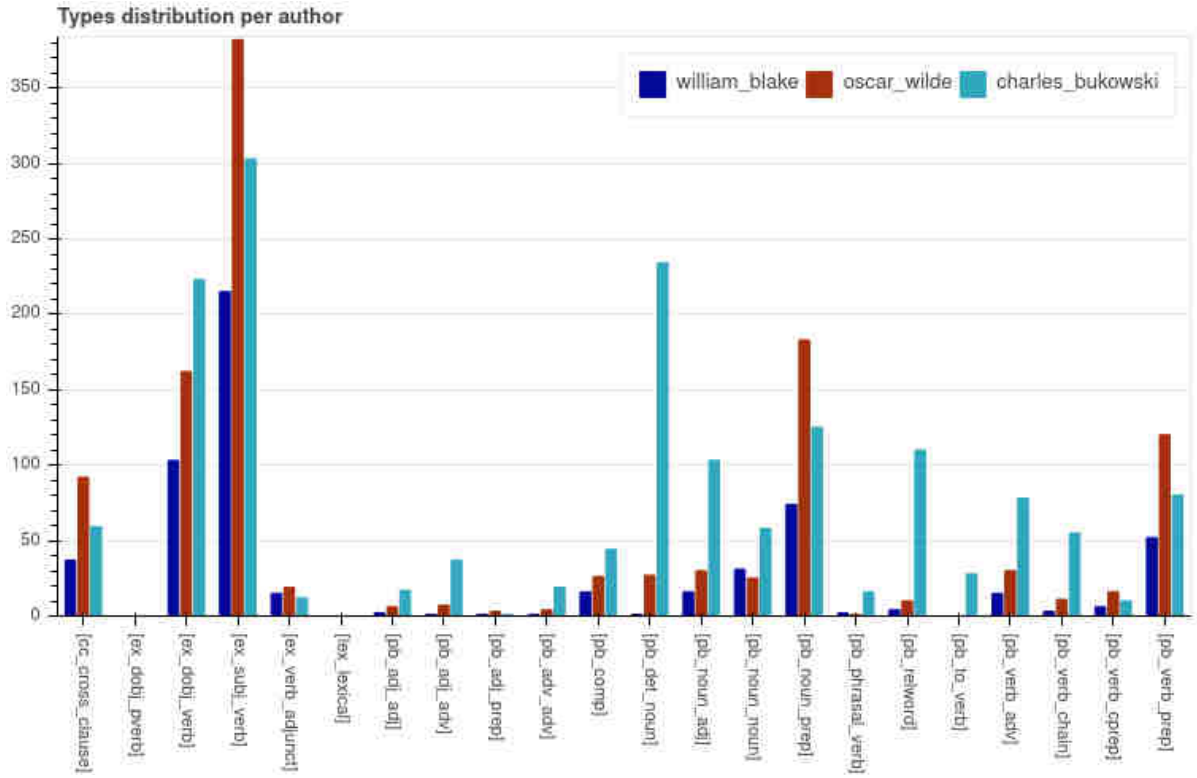

Figure 24: Distribution of types per author (Blake, Wilde, Bukowski)

Overall, Figure 24 shows that the same types are widely used by all three authors (expansions and enjambments between nouns and their complements). Yet, we can see that Wilde used some stronger or less conventional types which are more common in 20th poets, particularly *pb_det_noun, pb_adj_adj or pb_adv_adv*. We can also see that, in spite of being the most prone to use enjambments, Bukowski sometimes uses conventional types less than Wilde (cc_cross_clause, ex_subj_verb, pb_noun_prep).

The system's results reflect how earlier poets make less use of enjambment, given that enjambment was seen by some critics, especially Samuel Johnson, as disruptive of the line integrity (Hollander:91, 1975) and was thus used with restraint.

Diagrams displaying the distribution of enjambment for more authors can be found in Appendix 4.

8.4. Comparison between sonnets and non-sonnets

Looking at enjambment distribution in different poetic forms is also interesting. However, the overall corpus was not annotated with metadata, and the sonnets included in the reference corpus were chosen based on sonnets anthologies (some of them do not meet the usual sonnet layouts, such as "On the sale by auctions of Keats' love letters" by Oscar Wilde). Therefore, distinguishing between sonnets and non-sonnets in the corpus used in the case study is relatively hard. The heuristic applied was the following: if the poem was made of 14 lines and had rhymes, or the word "sonnet" was in its name, it was considered as a sonnet, otherwise, it was categorised as 'non-sonnet'.

Figure 25 below show the distribution obtained with this method compared to the reference corpus. As we can see observe, the distribution of types in the case study corpus, with the exception of *ex_dobj_verb* type, is also consistent with the distribution in the reference corpus.

JaDe's results identify a known trend for poems not conforming to the sonnet standards or poems from the 20th to have more contexts of enjambment than sonnets or earlier poems. Blank and free verses are more enjambment-friendly than sonnet lines as the latter are not bound by rhyming nor metric schemes and the former are not bound to rhyme. Poets from the 20th, especially modernist poets, did intend to depart from poetic conventions. However, the goal of this thesis, and especially of this section, is not to provide a literary analysis of the phenomenon, but only to give an overview of the kind of analysis JaDe could enable.

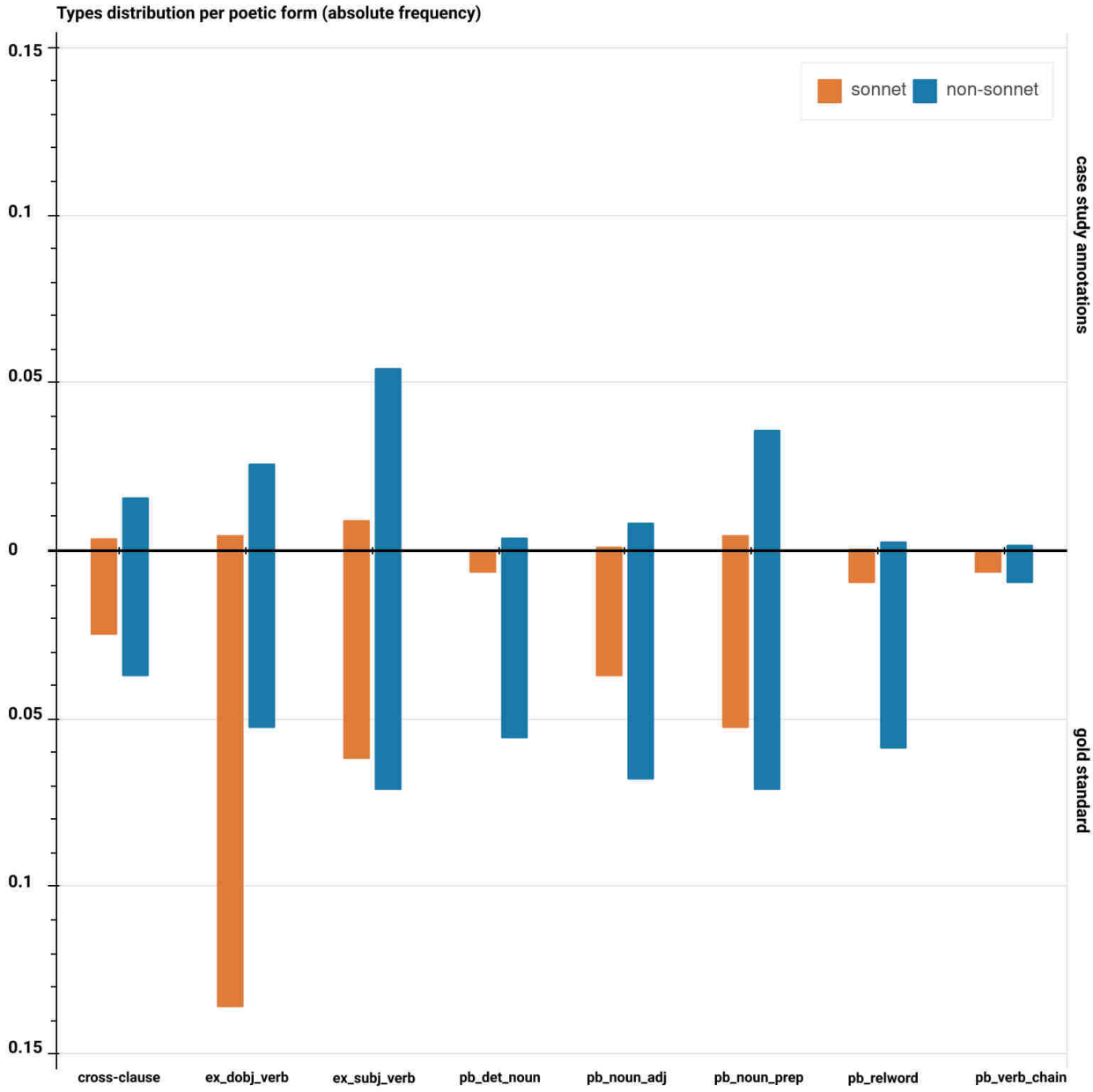

Figure 25: Distribution per poetic form of a few selected types in both the case study and reference corpora. Only the most reliable and the more common (in the reference corpus) types in the slightly less reliable ones are represented.

Conclusion and future work

This thesis consisted in the creation and evaluation of a tool to automatically detect enjambment in English poetry, along with the creation of the poetry corpora required for the task: a manually annotated reference corpus for the quantitative evaluation of the tool, and a larger corpus to assess the tool's applicability to literary analysis. Annotation guidelines were also created prior to reference corpus development.

The thesis began with a survey of different definitions and typologies of enjambment in order to provide the reader with a better understanding of the linguistic mechanisms at work, as well as contexts that could be challenging both for a human reader and a machine.

Out of the three typologies presented in section 1.3, Quilis and Spang's were adopted to automate the detection of enjambment. These typologies rely on syntactic cues and grammatical functions and have been previously used for enjambment detection in Spanish, which is the language they were originally intended for. They required a slight adaptation in order to better fit English syntax.

Our adaptation of the typologies to English was guided by corpus analysis: a corpus of poems was collected and annotated. The annotation first took place according to the types in the original typologies, keeping track of contexts that were not dealt with. The analysis of these contexts was then used to revise the typology for English, and the corpus was re-annotated with the revised typology.

JaDe, the system developed in the course of the thesis, is a rule-based system which works using morphological and syntactic cues, such as the part of speech or dependency relations of the words involved in the enjambments. A small amount of lexical knowledge was also integrated to handle phrasal verbs. The simple system thus created covers most enjambment contexts.

The system's goal is to assist literary researchers in their study of enjambments. The evaluation of the system showed that the system is able to detect the presence or absence of enjambment rather well (> 0.85 F1). However, it also showed that the results were mixed for the classification task, with F1 scores above 0.7 for some types, but with poorer scores for other types. Focusing mainly on types whose detection is more reliable, a case-study was carried out, applying the tool to a large corpus of poems from the 18th century to the 20th century. The case study highlighted patterns that correlate with already known facts, such as the increasing use of enjambment in later centuries. This can be seen as an indication that the tool can yield valid quantitative results about enjambment. We also saw how the tool can be used to quantify the use of more conventional and more rare enjambment types in different authors.

The tool could be improved in several ways. First, a syntactic typology was used, but other typologies have been proposed in the literature. In particular, Golomb speaks of prospective and retrospective enjambment. In prospective enjambment, the line is syntactically incomplete, and therefore enjambment is predictable. In retrospective enjambment, the line is semantically incomplete, and the reader does not become aware of this until they read the following line. The reference corpus was originally annotated with this typology as well: while we established that some syntactic types can be systematically associated with prospective or retrospective enjambment, most of them cannot. Given the lack of universal rules to classify them, they were left out of the system. Nonetheless, identifying the semantic types may highly benefit from resources like dictionaries (especially for verbs that can be both transitive and intransitive) or machine learning techniques.

The syntactic typology we used could also be improved in itself. In our corpus analysis, we found imprecisions in the existing definitions of enjambment. We also found contexts where it is unclear whether enjambment takes place or not. It would be possible to disambiguate these unclear cases

by interviewing native speakers of English, like Quilis did in his study with Spanish speakers, so as to ascertain whether or not a pause between the grammatical units involved is natural.

An analysis of errors made by the system showed that, in some cases, the sole use of POS and dependencies is not enough to distinguish between two enjambment types that have similar characteristics, such as *ex_verb_adjunct* and *pb_verb_prep*: In such cases, the use of semantic role labelling could help to avoid misclassification.

Error analysis for dependency-based classification highlighted the limits of mainstream NLP pipelines such as spaCy when working with poetic language. The linguistic specifics of poetry, including its non-standard spelling and casing, are two of the main limitations that a system focusing on poetic texts may encounter. Working on or testing an NLP pipeline trained on poetic or more literary texts could be explored for future work. Better corpus preprocessing could also be explored. In this work, poems were lowercased before being processed by the pipeline, however, truecasing could be tested in the future.

Furthermore, JaDe does not take into account the interaction between words and metrics. Involving metrics in the detection of enjambment is something that could be done in the future to improve the tool.

Finally, as of this writing, JaDe is a command line interface tool. For a tool intended to help literary researchers gather empirical proof, who may not be familiar with Python or the command line, this is not the best format of distribution. A web application would be an improvement towards this goal, and could include a platform to build diagrams from the annotations made by the tool.

References

Agirrezabal, M. (2017). Automatic Scansion of Poetry. San Sebastián/Donosti, Universidad del País Vasco.

Agirrezabal, M., Alegria, I., & Hulden, M. (2016, December). Machine learning for metrical analysis of English poetry. *In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (pp. 772-781).

BNC2 POS-Tagging Guide. (n.d.). Retrieved 23 March 2019, from http://ucrel.lancs.ac.uk/bnc2/bnc2guide.htm

Carroll, J., Minnen, G., & Briscoe, T. (1999). Corpus annotation for parser evaluation. *arXiv preprint cs/9907013*.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement, 20*(1), 37-46.

Choi, J. D., Tetreault, J., & Stent, A. (2015, July). It depends: Dependency parser comparison using a web-based evaluation tool. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 387-396).

De Marneffe, M. C., MacCartney, B., & Manning, C. D. (2006, May). Generating typed dependency parses from phrase structure parses. In Lrec (Vol. 6, pp. 449-454).

De Marneffe, M. C., & Manning, C. D. (2016). *Stanford typed dependencies manual* (pp. 338-345). Technical report, Stanford University.

Delente, É. (2019). Le traitement automatique du rythme régulier. Un cas particulier: L'enjambement [Automatic processing of the regular rhythm: the case of enjambment]. In *Proceedings of Plotting Poetry*, Nancy, France.

Delente, E., & Renault, R. (2015, June). Traitement automatique des formes métriques des textes versifiés [Automatic processing of metric forms in versified texts]. *In Actes de la 22e conférence sur le Traitement Automatique des Langues Naturelles. Articles courts* (pp. 116-122).

Enjambment Types—Spanish enjambment. (n.d.). Retrieved 1 April 2019, from https://sites.google.com/site/spanishenjambment/enjambment-types

Fort, K. (2012). *Les ressources annotées, un enjeu pour l'analyse de contenu: vers une méthodologie de l'annotation manuelle de corpus* [Annotated resources, a challenge for content analysis: towards a methodology for manual annotation of corpora] (Doctoral dissertation).

Golomb, H. (1979). A syntactically-oriented approach to line-end intonational tension and enjambment: A theoretical outline. In *Enjambment in poetry: Language and verse interaction* (pp. 269–310). Porter Institute for Poetics and Semiotics, Tel-Aviv University.

Hollander, J. (1975). 'Sense variously drawn out': On English enjambment. In *Vision and Resonance* (pp. 91–116). Oxford University Press.

Houston, N. (2014). Enjambment and the Poetic Line: Towards a Computational Poetics. In *Digital Humanities*, Lausanne, Switzerland. http://dharchive.org/paper/DH2014/Paper-430.xml

Hussein, H., Meyer-Sickendiek, B., & Baumann, T. (2018). Automatic detection of enjambment in German readout poetry. In *Proceedings of the Speech Prosody*, Poznan, Poland. https://doi.org/10.21437/SpeechProsody.2018-67

Kao, J., & Jurafsky, D. (2012, June). A computational analysis of style, affect, and imagery in contemporary poetry. In *Proceedings of the NAACL-HLT 2012 workshop on computational linguistics for literature* (pp. 8-17).

Katsamanis, A., Black, M., Georgiou, P. G., Goldstein, L., & Narayanan, S. (2011, January). SailAlign: Robust long speech-text alignment. In *Proceedings of workshop on new tools and methods for very-large scale phonetics research*.

King, T. H., Crouch, R., Riezler, S., Dalrymple, M., & Kaplan, R. M. (2003). The PARC 700 dependency bank. In *Proceedings of 4th International Workshop on Linguistically Interpreted Corpora* (LINC-03) at EACL 2003.

Klein, D., & Manning, C. D. (2003, July). Accurate unlexicalized parsing. *In Proceedings of the 41st annual meeting of the association for computational linguistics* (pp. 423-430).

Koops van't Jagt, R., Hoeks, J. C., Dorleijn, G. J., & Hendriks, P. (2014). Look before you leap: How enjambment affects the processing of poetry. *Scientific Study of Literature*, 4(1), 3–24.

Lawler, J. G. (1978). Enjambement: A Structure of Transcendence. *College English*, 39(6), 725-737.

Literature Online. (n.d.). Retrieved 1 April 2019, from https://literature.proquest.com/marketing/index.jsp

Lorteau, E. (2016). Normalisation de l'écrit pour le dialecte alsacien. [Standardisation of written Alsatian] [Unpublished master thesis]. Université de Strasbourg.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014, June). The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations* (pp. 55-60). https://www.aclweb.org/anthology/P14-5010.pdf

Marcus, M., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. https://doi.org/10.21236/ADA273556

Marcus, R. W. E. H. M., Palmer, M., Ramshaw, R. B. S. P. L., & Xue, N. (2011) OntoNotes: A Large Training Corpus for Enhanced Processing.

Martínez Cantón, C., Ruiz Fabo, P., González-Blanco, E., Poibeau, T. (2019). Automatic enjambment detection as a new source of evidence in Spanish versification. In Bories, A-S., Purnelle, G. and Marchal, H. *Plotting Poetry: on Mechanically-Enhanced Reading*. Liège, Presses de l'Université de Liège.

Monget E. (2020). JaDe Annotation Guide for the detection of enjambment in English poetry. https://doi.org/10.5281/zenodo.3992703

Monget, E., & Ruiz Fabo, P. (2020). Opportunités et limites en stylistique computationnelle de la poésie: Détection automatique de l'enjambement en anglais [Opportunities and limitations in poetic computational stylistics: automatic detection of enjambment in English poetry]. In *Proceedings of Humanistica 2020*, Bordeaux, France. http://doi.org/10.5281/zenodo.3788108

Mulholland, M., & Quinn, J. (2013, October). Suicidal tendencies: The automatic classification of suicidal and non-suicidal lyricists using nlp. In *Proceedings of the sixth international joint conference on natural language processing* (pp. 680-684).

Natural Language Toolkit—NLTK 3.4 documentation. (n.d.). Retrieved 23 March 2019, from http://www.nltk.org/

Navarro-Colorado, B. (2018). A metrical scansion system for fixed-metre Spanish poetry. Digital Scholarship in the Humanities, 33(1), 112-127.

Nivre, J., De Marneffe, M. C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., Tsarfaty, R. (2016, May). Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation* (LREC'16) (pp. 1659-1666).

Oxford University Press (2020). sentence. In: *Lexico.com*, Available at: https://www.lexico.com/definition/sentence [Accessed 14/07/2020].

Parry, M. (1929, January). The distinctive character of enjambement in Homeric verse. In *Transactions and Proceedings of the American Philological Association* (Vol. 60, pp. 200-220). Johns Hopkins University Press, American Philological Association. https://doi.org/10.2307/282817

Pradhan, S., Moschitti, A., Xue, N., Ng, H. T., Björkelund, A., Uryupina, O., ... & Zhong, Z. (2013, August). Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning* (pp. 143-152).

Quilis, A. (1964). Estructura del encabalgamiento en la métrica espanola. [The Structure of Enjambement in Spanish Metrics]. Consejo Superior de Investigaciones Cientıficas, patronato Menéndez y Pelayo, Instituto Miguel de Cervantes.

Rafferty, A. N., & Manning, C. D. (2008, June). Parsing three German treebanks: Lexicalized and unlexicalized baselines. In *Proceedings of the Workshop on Parsing German* (pp. 40-46).

Ruiz Fabo, P., Bermúdez Sabel, H., Martínez Cantón, C. I., González-Blanco García, E., & Navarro Colorado, B. (2018). The diachronic Spanish sonnet corpus (DISCO): TEI and linked open data encoding, data distribution and metrical findings.

Ruiz, P., Martínez Cantón, C., Poibeau, T., & González-Blanco, E. (2017). Enjambment Detection in a Large Diachronic Corpus of Spanish Sonnets. Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature, 27–32. https://doi.org/10.18653/v1/W17-2204

Schiller, A., Teufel, T., Stöcker C., Thielen, C. (1999). Guidelines für das Tagging deutscher Textcorpora mit STTS (Kleines und größes Tagset) [Guidelines for tagging German textcorpora with STTS (small and large tagsets]. http://www.ims.uni-stuttgart.de/forschung/ressourcen/lexika/TagSets/stts-1999.pdf.

SpaCy · Industrial-strength Natural Language Processing in Python. (n.d.). Retrieved 17 March 2019, from https://spacy.io/

Spang, K. (1983). *Ritmo y versificación: teoría y práctica del análisis métrico y rítmico*. [Rhythm and versification: theory and practice of metric and rhythmic analysis]. Editum.

Stanford CoreNLP – Natural language software | Stanford CoreNLP. (n.d.). Retrieved 23 March 2019, from https://stanfordnlp.github.io/CoreNLP/

Tesnière, L. (1959). *Eléments de syntaxe structurale* [Elements of structural syntax]. Klincksieck.

Tourrette, É. (2004). Poète après tout. L'art de l'enjambement dans les Quatrains de Pibrac. [Poet after all. The art of enjambment in Pibrac's quatrains] *L'information Grammaticale*, 100(1), 4-8. https://doi.org/10.3406/igram.2004.2579

Weischedel, R., Hovy, E., Marcus, M., Palmer, M., Belvin, R., Pradhan, S., ... & Xue, N. (2011). OntoNotes: A large training corpus for enhanced processing. Handbook of Natural Language Processing and Machine Translation. Springer, 59.

Wesling, D. (1996). The Scissors of Meter: Grammetrics and Reading. University of Michigan Press.

Wolfson, S. J. (2003). Blake's language in poetic form. In *The Cambridge companion to William Blake* (pp. 63–84). Cambridge University Press.

Appendices

Appendix 1. Phrase-bounded types with examples in English and French

NP/Adj	Break up of a noun and the adjective modifying it	"D'une **langueur/Monotone**" Paul Verlaine, "Chanson d'automne"	"a red wheel/Barrow/blazed with rain/water/beside the **white/Chickens"** William CW, "The Red Wheelbarrow"
PP/Adj	Break up of a adjective and prepositional phrase modifying it	"et mon coeur, **transparent/Pour elle seule**, hélas ! cesse d'être un problème" Paul Verlaine, "Mon rêve familier"	"is it still not **possible/to** die for somebody else?" Charles Bukowski, "On The Fire Suicides Of The Buddhists "
Adv/Adj	Break up of an adjective and the adverb modifying it		"But a silence vasty-deep, oh **deeper** than all these ties/**Now**" Claude McKay, "Absence"
Adv/PP	Break up of a prepositional phrase and the adverb modifying it		"crawling in and **out/of** beds." Charles Bukowski, "Alone with everybody"
NP/PP	Break up of a noun and the relative prepositional phrase modifying it	"Nous sommes **les Ingénues/Aux bandeaux plats**, à l'oeil bleu" Paul Verlaine, "La chanson des ingénues"	"I saw a **girl with one leg/over the rail of a balcony**" Williams CW, "The Right Of Way"
Relword	Break up a lexical word and the grammatical word (determiner, conjunction...) related to it	"Se levant blafarde et solennelle, **une/Nuit** mélancolique et lourde d'été" Paul Verlaine, "Le rossignol"	"for more **than/flesh**." Charles Bukowski, "Alone with Everybody"
VP/Adv	Break up of a verb and the adverb modifying it		"Its loveliness increases; it will **never/Pass** into nothingness; but still will keep" John Keats, "Endymion"
VP/Prep	Break up of a verb and the prepositional phrase modifying it	"Et la nuit terne arrive et Vénus **se balance/Sur une molle nue** au fond des cieux obscurs" Paul Verlaine, "Nocturne Parisien"	"The free bird **leaps/on** the back of the wind" Maya Angelou, "Caged Bird"
V_chain	Break up of a phrasal verb or of a verb and its auxiliary	"wept for the lovers who **had/hurt** and forgotten you." Charles Bukowski, "An almost made up poem"	

Appendix 2. Annotation guide

Syntactic annotations

Tag	Description	Example
Lexical Type		
lex_lexical	Word spread onto two lines because of the enjambment	a red wheel barrow William Carlos Williams, “The Red Wheelbarrow”
Cross-clause Type		
cc_cross_clause	Enjambment occurs between a noun and the relative modifying that noun	Life is a broken-winged bird that cannot fly Langston Hughes, “Dreams”
Phrase-bounded Types		
pb_verb_chain	Enjambment occurs between a verb and its auxiliary or modal	the lovers who had hurt and forgotten you Charles Bukowski, “An Almost Made Up Poem”
pb_verb_cprep	Enjambment occurs between a verb and its lexically-related preposition introducing a prepositional phrase	still, I thought somehow you'd like to know? Charles Bukowski, “Hemingway Never Did That”
pb_verb_prep	Enjambment occurs between a verb and the prepositional phrase modifying it	The free bird leaps on the back of the wind Maya Angelou, “The Caged Bird”
pb_verb_adv	Enjambment occurs between a verb and the adverb modifying that verb	Its loveliness increases;it will never Pass into nothingness John Keats, “Endymion”
pb_noun_adj	Enjambment occurs between and the adjective modifying that noun	beside the white chickens William Carlos Williams, “The Red Wheelbarrow”
pb_noun_prep	Enjambment occurs between and the prepositional phrase modifying that noun	I saw a girl with one leg over the rail of a balcony William Carlos Williams, “The right of Way”
pb_noun_noun	Enjambment occurs between the two nouns of a compound noun	Alone in my purity World without end Langston Hughes, “God”

Tag	Description	Example
pb_det_noun	Enjambment occurs between a noun and the determiner related to that noun	you were one of the best female poets Charles Bukowski, “An Almost Made Up Poem”
pb_adj_adv	Enjambment occurs between an adjective and the adverb modifying that adjective	Is it still possible to die for someone else
pb_adj_prep	Enjambment occurs between an adjective and a prepositional phrase modifying that adjective	is it still possible to die for someone else Charles Bukowski, “On the Fire Suicides of the Buddhists”
pb_adj_adj	Enjambment occurs between two adjectives modifying the same noun	I was at that ripe old age of 45 Charles Bukowski, “Trollius and Trellises”
pb_adv_adv	Enjambment occurs between two adverbs related to the same verb	grabs him and pulls him back down Charles Bukowski, “The Great Escape”
pb_phrasal_verb	Enjambment occurs between a phrasal verb and the preposition/particle related to that phrasal verb	I won’t blame him for getting Out Charles Bukowski, “Trollius and Trellises”
pb_to_verb	Enjambment occurs between a ‘to’ preposition and the verb it introduces	We began our unholy alliance to test the literary waters Charles Bukowski, “Trollius and Trellises”
pb_comp	Enjambment occurs between elements that are part of a comparison structure	Better to be human Than God Langston Hughes, “God”
pb_relword	Enjambment occurs between a conjunction and the clause it introduces	You wrote me that letter and I answered Charles Bukowski, “An Almost Made Up Poem”
Expansion Type		
ex_subj_verb	Enjambment occurs between a subject and its governing verb	[…] all in upper case, and you knew famous artists Charles Bukowski, “An Almost Made Up Poem”

Tag	Description	Example
ex_dobj_verb	Enjambment occurs between a direct object and its governing verb	her dressing did outbrave all the pride the fields then have Ben Jonson, Underwoods, “How he saw her”
ex_dobj_pverb	Enjambment occurs between a phrasal verb and its direct object	haven’t you heard of the thirties? Charles Bukowski, “Trollius and Trellises”
ex_verb_adjunct	Enjambment occurs between an adverbial clause and its governing verb	And then awakens in the morning to write Charles Bukowski, “An Almost Made Up Poem”

In case of ex_verb_adjunct, the type of adjunct should be specified (time, location, manner…) :

1. Afterward I went past what you had passed
2. Before we met and you what I had passed.

01 02 [ex_verb_adjunct] [suave] [retro] [time]

Style annotation

Tag	Description	Example
abrupt	The meaning of the enjambed sentence flows over the next line but there is some sort of pause occurring somewhere before the end of that line as well	I am not prone to weeping, as our sex Commonly are; the want of which vain dew Perchance shall dry your pities; but I have [that honourable...] William Shakespeare, “The Winter’s Tale”
smooth	The meaning of the enjambed sentence flows over the next line and no pause occurs before the end of that line	Hold fast to dreams For if dreams die Life is a broken-winged bird That cannot fly. Langston Hughes, “Dreams”

Meaning annotation

Tag	Description	Example
retro	The line or sentence is syntactically complete but the following line contains semantic information necessary to understand the sentence right	Life is a barren field Frozen with snow Langston Hughes, “Dreams”
pro	The line or sentence is syntactically incomplete and must be completed by the following line	Below me young lovers Tread the sweet ground Langston Hughes, “God”

Example

1. A thing of beauty is a joy for ever:
2. Its loveliness increases; it will never
3. Pass into nothingness; but still will keep
4. A bower quiet for us, and a sleep
5. Full of sweet dreams, and health, and quiet breathing.

01 02 [] [] []

02 03 [pb_verb_adv] [abrupt] [pro]

03 04 [ex_dobj_verb] [abrupt] [pro]

04 05 [pb_noun_adj] [abrupt] [pro]

Appendix 3. Examples of annotated poems

Admonition - Sylvia Plath
1. If you dissect a bird
2. To diagram the tongue
3. You'll cut the chord
4. Articulating song.

5. If you flay a beast
6. To marvel at the mane
7. You'll wreck the rest
8. From which the fur began.

9. If you pluck out the heart
10. To find what makes it move,
11. You'll halt the clock
12. That syncopates our love.

01 02 [ex_verb_adjunct] [smooth] [retro]
02 03 [] [] []
03 04 [cc_cross_clause] [smooth] [retro]
04 05 [] [] []

05 06 [ex_verb_adjunct] [smooth] [retro]
06 07 [] [] []
07 08 [pb_noun_prep] [smooth] [retro]
08 09 [] [] []

09 10 [ex_verb_adjunct] [smooth] [retro]
10 11 [] [] []
11 12 [cc_cross_clause] [smooth] [retro]

Dreams - Langston Hughes
1. Hold fast to dreams
2. For if dreams die
3. Life is a broken-winged bird
4. That cannot fly.
5. Hold fast to dreams
6. For when dreams go
7. Life is a barren field
8. Frozen with snow.

01 02 [] [] []
02 03 [] [] []
03 04 [cc_cross_clause] [smooth] [retro]
04 05 [] [] []
05 06 [] [] []
06 07 [] [] []
07 08 [pb_noun_adj] [smooth] [retro]

Appendix 4. Diagrams - Enjambment distribution per author in the case-study corpus

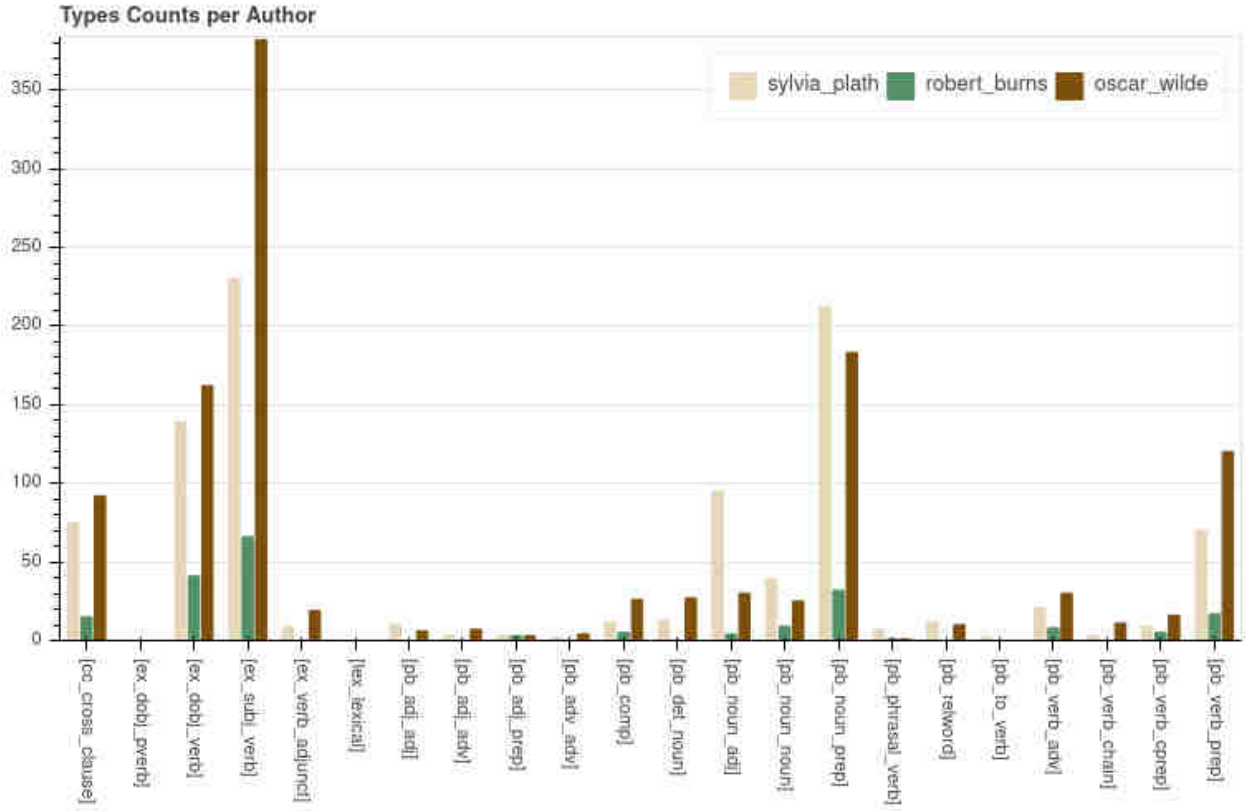

Figure 26: Comparison of authors' use of enjambment (a)

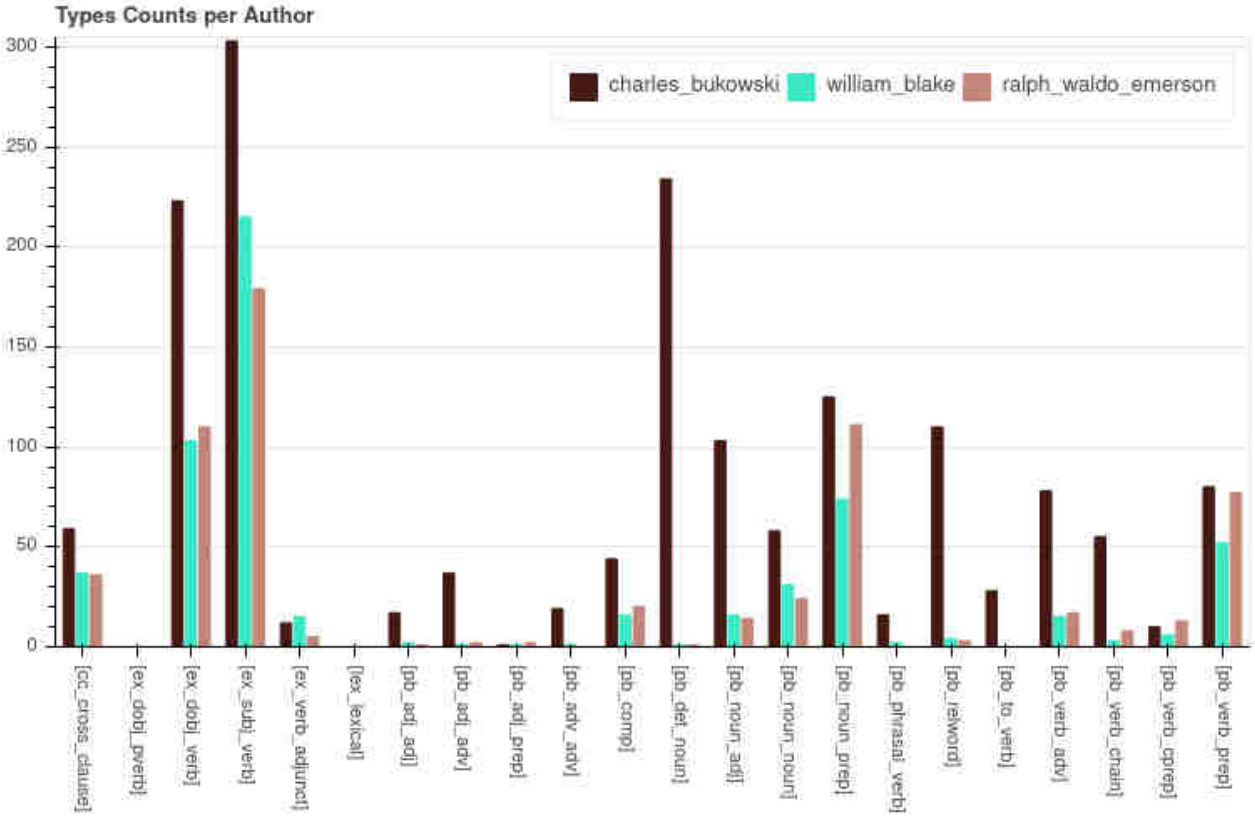

Figure 27: Comparison of authors' use of enjambment (b)

Appendix 5. Diagrams - comparison of the distribution of the most reliable and frequent enjambment types, in both system and golden annotations (per period)

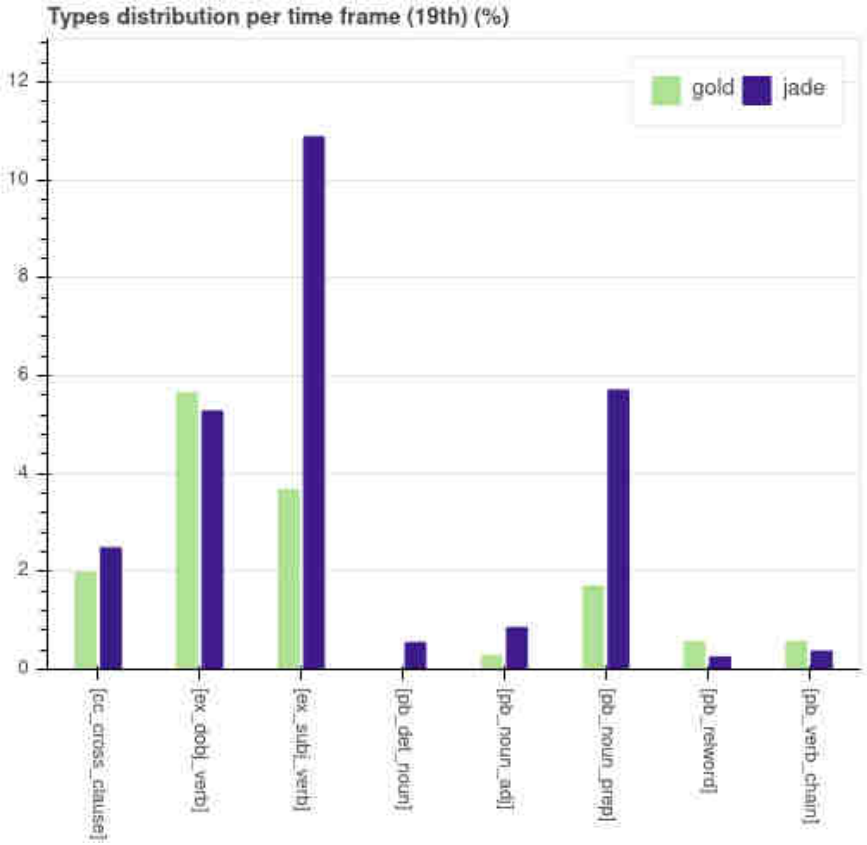

Figure 28: Comparison of the distribution (expressed in percentage) of the most reliable and frequent enjambment types in both system and golden annotation (19th)

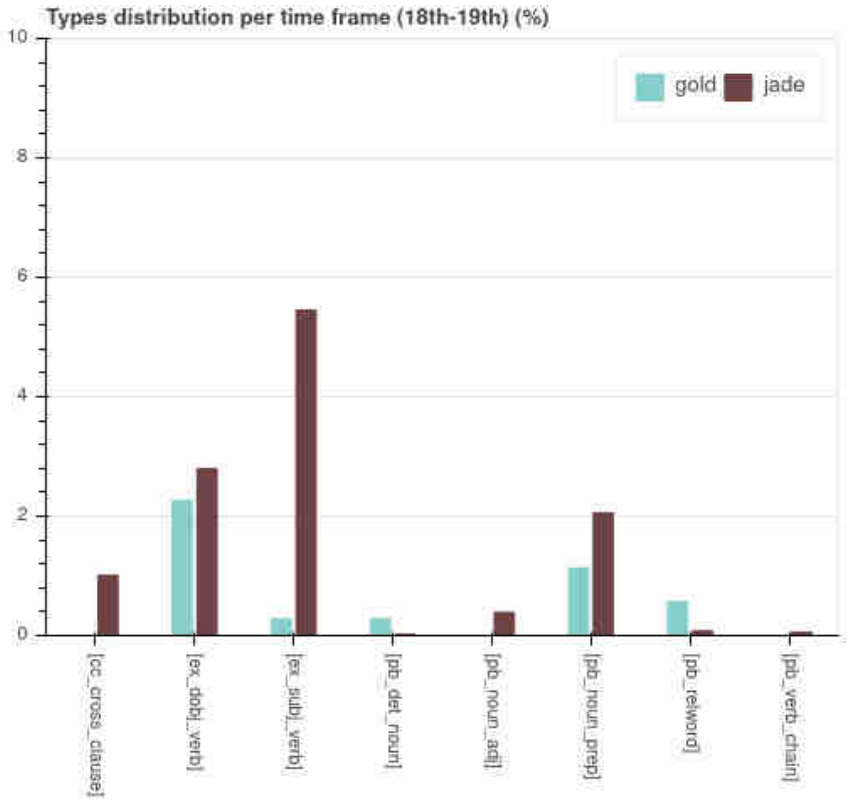

Figure 29: Comparison of the distribution (expressed in percentage) of the most reliable and frequent enjambment types in both system and golden annotation (18th-19th)

Appendix 6. Chart maker - a Bokeh wrapper developed for enjambment analysis

With the exception of Figures 21 and 25, which were engineered for the comparison to be easier to read, every diagram in sections 5, 7, 8 and appendices 4 and 5, were built using bokeh,[42] a python library dedicated to plotting. Chart maker (Charmak), created in the course of this thesis, is a small package wrapped around bokeh which helps building 4 different types of diagrams. Instead of manipulating the bokeh objects, the user only calls the function corresponding to the type of diagrams they want to build. These functions allow for an easy customisation of plots, including taking care of the number of colours to be used according to the number of *x* to be displayed (although there is currently no control over the colours to be generated, which is of course a limitation).

Charmak[43] and the functions[44] used to create the data structure to build the plots are to be made available along with JaDe's source code. The code below shows a Charmak example.

```
def stacked_plot(x, y, data, title, x_name, orientation='horizontal',
                 plot_width=750, height=750, colors=None, y_start=0,
                 location='top_left', bar_width=0.5, save=False,
                 save_name=None):

    if colors is None:
        colors = random_colors(len(data))

        tooltips = f'$name @{x_name}: @$name'

         stacked_figure = figure(x_range=x, plot_height=height,
                                 plot_width=plot_width, title=title,
                                 toolbar_location=None, tools="hover",
                                 tooltips=tooltips)

        stacked_figure.vbar_stack(y,x=x_name,width=bar_width,
        color=colors, source=data, legend_label=y)

        stacked_figure.y_range.start = 0
        stacked_figure.x_range.range_padding = 0.1
        stacked_figure.legend.location = location
        stacked_figure.legend.orientation = orientation
        stacked_figure.xaxis.major_label_orientation = orientation

    if save or save_name is not None:
        output_file(save_name)

    return stacked_figure
```

Figure 30: One of charmak's function: s*tacked figure* refers to the figure visible in example `bar_stacked.py` in Bokeh's documentation: https://docs.bokeh.org/en/latest/docs/gallery/bar_stacked.html

[42] https://docs.bokeh.org/en/2.1.1/
[43] https://github.com/MongetE/JaDe/tree/master/JaDe/jane/charmak
[44] https://github.com/MongetE/JaDe/blob/master/JaDe/jane/run.py; https://github.com/MongetE/JaDe/blob/master/JaDe/jane/jane.ipynb