FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of bachelor's thesis

Title:	Automatic poetic metre detection
Student:	Kristýna Klesnilová
Supervisor:	Ing. Karel Klouda, Ph.D.
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2022/2023

Instructions

This work aims to create a tool that would decide for a poem written in Czech what metre it is written in. This task can be divided into three steps:
1. Detection of (phonetic) syllable boundaries.
2. Deciding whether a given syllable is an accented or long syllable.
3. Assigning a metre to the given poem.

For each of these steps, perform a search for existing solutions and, if necessary, try to find a solution via machine learning using the dataset [1]. Compare the resulting models with existing tools and evaluate their performance.

[1] Plecháč, Petr – Ibrahim, Robert (2013). Databáze českých meter. Praha: Ústav pro českou literaturu AV ČR.

Electronically approved by Ing. Karel Klouda, Ph.D. on 26 January 2022 in Prague.

Bachelor's thesis

AUTOMATIC POETIC METRE DETECTION

Kristýna Klesnilová

Faculty of Information Technology
Department of Applied Mathematics
Supervisor: Ing. Karel Klouda, Ph.D.
May 11, 2022

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis: Klesnilová Kristýna. *Automatic poetic metre detection.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2022.

Contents

List of Figures

List of Tables

In the first place, I would like to express sincere gratitude to my supervisor, Ing. Karel Klouda, Ph.D., for all his excellent advice and constant guidance in writing this thesis. Next, I would like to thank my family and friends for their support during my studies. Especially to all the amazing friends from CHS, who made my studies much more fun.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on May 11, 2022 . . . .

Abstract

This work is devoted to automatic metrical analysis of Czech syllabotonic verse metrically tagged inside a large poetic corpus – the Corpus of Czech Verse. First, it reimplements the existing data-driven approach used by a program called KVĚTA. Later, it models the problem as a sequence tagging task and solves it using machine learning. The BiLSTM-CRF model is used, representing the current state of the art for many sequence tagging tasks. Many different input configurations are tested. In all experiments, the inputted syllables or word tokens are represented by Word2Vec word embeddings trained on training data. The results are evaluated by computing three different accuracies of the predictions: syllable-level accuracy, line-level accuracy, and poem-level accuracy. It is shown that using BiLSTM-CRF represents a great success. With the best input configurations, it produces better results than the KVĚTA reimplementation, with predictions achieving 99.61% syllable accuracy, 98.86% line accuracy, and 90.40% poem accuracy. The most interesting finding is that the best results are obtained by inputting sequences representing whole poems instead of individual poem lines.

Keywords automatic metrical analysis of verse, Czech syllabotonic verse, Corpus of Czech Verse, KVĚTA, BiLSTM-CRF, Word2Vec

Abstrakt

Tato práce se zabývá automatickou metrickou analýzou českého sylabotonického verše, jenž je metricky otagován ve velkém korpusu básní – v Korpusu českého verše. Práce nejprve reimplementuje přístup založený na datech, který využívá program s názvem KVĚTA. Poté si metrickou analýzu namodeluje jako úlohu tagování sekvencí a řeší ji pomocí strojového učení. Je trénován model BiLSTM-CRF, který reprezentuje aktuálně nejlepší architekturu pro většinu klasických úloh tagování sekvencí. Je otestováno mnoho různých vstupních konfigurací. Ve všech experimentech jsou slabiky nebo tokeny slov na vstupu reprezentovány pomocí Word2Vec embeddingů natrénovaných na trénovacích datech. Výsledky jsou vyhodnoceny pomocí spočítání tří různých přesností predikce: přesnosti pro jednotlivé slabiky, přesnosti pro jednotlivé řádky básní a přesnosti pro celé básně. Je ukázáno, že použití modelu BiLSTM-CRF představuje velký úspěch. S nejlepšími vstupními konfiguracemi vrací BiLSTM-CRF lepší výsledky než reimplementace programu KVĚTA s predikcemi dosahujícími 99.61% přesnosti pro jednotlivé slabiky, 98.86% přesnosti pro jednotlivé řádky básní a 90.40% přesnosti pro celé básně. Nejzajímavější zjištění představuje fakt, že nejlepších výsledků je dosaženo pro vstupní sekvence reprezentují celé básně namísto jednotlivých řádků básní.

Klíčová slova automatická metrická analýza verše, český sylabotonický verš, Korpus českého verše, KVĚTA, BiLSTM-CRF, Word2Vec

List of abbreviations

POS	Part of speech
NER	Named entity recognition
RNN	Recurrent neural network
LSTM	Long short-term memory
BiLSTM	Bidirectional LSTM
CRF	Conditional random fields
LSTM-CRF	LSTM network with a CRF layer
BiLSTM-CRF	Bidirectional LSTM network with a CRF layer
CNN	Convolutional neural network
SGD	Stochastic gradient descent
HMM	Hidden Markov model
MPP	Monosyllabic preposition proper

Introduction

Metrical analysis of verse is an important versology task that consists of analysing a poem and deciding in which metre it is written. This task is challenging not only because of the complexity of its subtasks (splitting a word into syllables, deciding whether a syllable is accented or long ...) but also due to the fact that poets – as all artists – tend to be creative and do not always follow the metrical norms precisely. Therefore, all algorithms and proposed approaches must take these "creative mistakes" into account.

In the past, metrical analysis had to be performed only algorithmically using rule-based approaches. Nowadays, there exist large corpora containing many semi-automatically tagged poems, and thanks to that, data-driven approaches are possible. In this work, one such corpus is used – the Corpus of Czech Verse [1]. This work reimplements the statistical approach to the metrical analysis of Czech syllabotonic verse as performed within the Czech verse processing system KVĚTA, which was developed by the authors of the Corpus of Czech Verse. Afterwards, the task is modelled as a sequence tagging task, and further experiments are performed using a state-of-the-art machine learning approach – the BiLSTM-CRF model for sequence tagging. The use of this model has been recently proposed by versology researchers [2], but it has not yet been tested on the Czech syllabotonic verses inside the Corpus of Czech Verse.

Motivation

The results of this thesis will be beneficial to versology researchers, as the BiLSTM-CRF model is used for the first time with Czech syllabotonic verse. If this work proposes some new input configurations for the BiLSTM-CRF model, the configurations may be beneficial even for researchers working with verses written in other languages.

Thesis structure

This work begins with a theoretical background. It introduces necessary concepts from the theory of verse and machine learning (Chapters 1 and 3), describes the Corpus of Czech Verse (Chapter 2) and presents the metrical analysis pipeline with all its subtasks and possible approaches to solve them (Chapter 4). Later, it continues with a practical part, where the reimplementation of the KVĚTA program and training of the BiLSTM-CRF model with various input configurations are described (Chapter 5). Finally, the obtained results are presented and discussed (Chapter 6).

Objectives

The theoretical part of this work aims to build common ground and introduce the reader to the concepts of verse and machine learning theory used in this thesis. Furthermore, it intends to present the structure and contents of the Corpus of Czech Verse, the metrical tagging pipeline with all its subtasks, and the existing approaches to solve this task.

The objective of the practical part is to reimplement the KVĚTA data-driven approach and train the BiLSTM-CRF sequence tagging model. For the BiLSTM-CRF, the goal is to propose various input configurations that may be beneficial to the model and to test all of them. Based on the obtained results, the aim is to decide whether using the BiLSTM-CRF model for the metrical tagging of Czech syllabotonic verse is successful and has some benefits over using the KVĚTA approach.

Chapter 1

Verse theory

This chapter introduces essential concepts from verse theory that are used throughout this thesis and are vital for a better understanding of the rest of this work.

1.1 Poem structure

1.1.1 Verse

Poetry is written in verse. A common misconception is that one verse is equivalent to one line in a poem. That is not true in all cases. The verse does not have to end with a line end, but it can be spread across multiple lines. This often occurs, for example, inside dialogues in verse dramas. When a verse does not end with a line end, it is called an *enjambement.* [3]

For an example of an enjambement, see Figure 1.1.

Dí paní domu; dítě přiblíží
se těsně k ní, vytáhne z haleny
list složený...

Figure 1.1 Enjambement [4]

1.1.2 Strophe

Verses inside a poem can be organised into strophes. A strophe is a group of verses that form a semantic unit and tend to be graphically separated. The strophe is then repeated throughout the poem with similar or almost similar properties (same number of verses, same metrical or rhyme scheme). Strophes tend to contain 2–14 verses – strophes with an even number of verses are more common. When shorter and longer verses alternate in a strophe, then usually in such an order that a longer verse comes before a shorter one. The last verse of a strophe tends to be shorter or less rhythmically regular. [3] Some strophes even have special names assigned to them, for example, *the Sapphic stanza*, *the Alcaic stanza*, *the Second Asclepiad stanza*, or *the Fourth Asclepiad stanza.* [4]

1.1.3 Paragraph (Stanza)

However, in many poems, groups of verses can be encountered that are graphically separated; nevertheless, their internal organisation lacks any regularity. These are not called strophes but paragraphs (stanzas). [3]

1.2 Versification systems

This section introduces four important versification systems – *syllabic*, *quantitative*, *tonic*, and *syllabotonic* – along with the different approaches towards versification. However, only syllabotonic versification will be further discussed in the rest of this work, as within the Corpus of Czech Verse, metres are assigned only to syllabotonic verses. [1]

1.2.1 Syllabic versification system

Probably the oldest versification system used in Indo-European languages, syllabic, distinguishes verses only by the number of syllables they contain. Syllabic versification does not care whether the syllables are accented or long, the only important thing is their number. [4] In Czech poetry, the syllabic verse was used until the end of the 18th century. After that, it appeared only episodically, especially in folk poetry. [3]

In the excerpt from the poem *Co Bůh? Člověk?* by Fridrich Bridel (see Figure 1.2), no regularity can be found in the alternation of long and short syllables or the accented and non-accented syllables. However, it can be noted that 7-syllable and 8-syllable verses regularly alternate and verses with the same number of syllables rhyme. Therefore, it represents a syllabic versification. [3]

Ja|ký | boj? | Ja|ké | hnu|tí? (7 syll.)
mně | vstu|pu|jí | na | myš|le|ní? (8 syll.)
Mám|-li | snad | za|hy|nou|ti? (7 syll.)
Či|ji | mdlé | při|ro|ze|ní. (7 syll.)
Což | to? | Věc | vel|mi | rych|lá, (7 syll.)
a|neb | jest|-li | ňá|ké | zdá|ní, (8 syll.)
ros|tou | mně | ja|kás | kří|dla, (7 syll.)
stro|jí | se | vše|cko | k lí|tá|ní. (8 syll.)

■ **Figure 1.2** Syllabic versification

1.2.2 Quantitative versification system

The quantitative versification system differentiates between long and short syllables. The long and short syllables are annotated according to the rules that take diphthongs, vocals, and syllable-forming consonants into account. Quantitative verse can be found in Greek and Roman poetry. In Czech poetry, it was used mainly in the 16th century and then shortly in the 1920s.

In the quantitative versification system, a syllable can be long by nature or long by position. Syllable long by nature contains a long vocal or a diphthong. Meanwhile, a syllable long by position contains either a short vocal or a syllable-forming consonant *l* or *r*. This short vocal or syllable-forming consonant is followed by two or more consonants (not necessarily belonging to the same syllable). On the other hand, when a syllable contains a short vocal or a syllable-forming consonant followed by only one consonant, the syllable is classified as short.

When a syllable contains a short vocal or a syllable-forming consonant followed by exactly two consonants and one of the two consonants is *l*, *r*, *ř*, *m* or *n*, the syllable can be long or short depending on the context.

In the passage from the poem *Noční bdění* by I. V. Šimko (see Figure 1.3), all long syllables occupy an odd position – except for the second position in the last verse and the final positions in all verses except the last one. The final position in verse represented a common exception and could be occupied by both long and short syllables. Therefore, the poem is classified as quantitative. [3]

sen | mi|lý | po|koj|ně | lí|tá,
a | v le|sích | zpě|vák | mi|lost|ný
s pří|ro|dou | ce|lou | spo|čí|vá:
teh|dy | já | se | mar|ně | trá|pím
blou|dě | v há|ji | až | do | rá|na

Figure 1.3 Quantitative versification (long syllables are underlined)

1.2.3 Tonic versification system

Tonic versification was the second most important versification system in medieval Europe. The tonic verse normalises the number of accents in a line. Usage in Czech poetry is very rare, oftentimes readers confuse it with free verse [4] (verse without a metrical norm [3]).

In the example of a tonic poetic text (see Figure 1.4), every verse has exactly four accents, but the syllable counts differ. [4]

A | vrá|til | se | Mu|ro|mec | k dob|ré|mu | mlád|ci, (4 acc., 12 syll.)
K mlád|ci | to|mu | dob|ré|mu, | u|bi|té|mu; (4 acc., 11 syll.)
On | vy|ko|pal | hrob | v ší|rém | po|li, (4 acc., 9 syll.)
Do | to|ho | hro|bu | tě|lo | po|lo|žil (4 acc., 10 syll.)

Figure 1.4 Tonic versification (accented syllables are underlined)

1.2.4 Syllabotonic versification system

Syllabotonic versification combines syllabic and tonic versification considering not only the number of syllables but also whether they are accented or not. [4]

In the excerpt from the poem *U studánky* by Jan Neruda (see Figure 1.5), every accented syllable, except the second syllable in the fourth verse, occupies an odd position in verse. Furthermore, every verse has exactly eight syllables. Therefore, syllabotonic versification is used. [3]

In the rest of this work, the Czech syllabotonic verse will be discussed.

U | stu|dá|nky | sto|jí | děv|če, (8 syll.)
mla|dé | ja|ko | strů|mek | mla|dý, (8 syll.)
ble|dé | ja|ko | ru|báš | z kmen|tu. (8 syll.)
A | na | ne|bi | bí|lý | mě|síc, (8 syll.)
ko|lem | ně|ho | vod|ní | ko|lo (8 syll.)
jak | by | ze | stu|dá|nky | hle|děl. (8 syll.)

Figure 1.5 Syllabotonic versification (accented syllables are underlined)

1.3 Metrical analysis properties

When performing a metrical analysis of a poem, various properties of the verse can be examined.

1.3.1 Foot

The basic metrical unit of a verse is called a foot. In the syllabotonic verse, it represents a group of at least two syllables that is repeated regularly throughout the verse. One foot consists of strong and weak positions. Strong positions are labelled with **S**, and weak positions are labelled with **W**. If there are two weak positions within a foot, the first is labelled using **V**. Table 1.1 presents all types of feet that can be encountered within the Czech syllabotonic verse. [4]

Table 1.1 Czech syllabotonic verse feet (positions inside brackets can be omitted)

Foot	Feet pattern
Iamb	$W_0\ S_1\ W_1\ S_2\ \ldots\ S_n\ (W_n)$
Trochee	$S_1\ W_1\ S_2\ W_2\ \ldots\ S_n\ (W_n)$
Dactyl	$S_1\ V_1\ W_1\ S_2\ V_2\ W_2\ \ldots\ S_n\ ((V_n)\ W_n)$
Dactyl with anacrusis (Amphibrach)	$W_0\ S_1\ V_1\ W_1\ S_2\ V_2\ W_2\ \ldots\ S_n\ ((V_n)\ W_n)$
Dactylotrochee	$S_1\ (V_1)\ W_1\ S_2\ (V_2)\ W_2\ \ldots\ S_n\ ((V_n)\ W_n)$
Dactylotrochee with anacrusis	$W_0\ S_1\ (V_1)\ W_1\ S_2\ (V_2)\ W_2\ \ldots\ S_n\ ((V_n)\ W_n)$

All standard syllabotonic metrical patterns can be expressed by the following regular expression:

$$\texttt{\^{}W?(SWW?)*(SW?)?\$} \tag{1.1}$$

where V and W weak positions are annotated with the same symbol. [5]

It is important to note that foot and word are two different concepts. Their boundaries do not have to overlap. Two different situations are distinguished:

Caesura The word does not end where the foot ends.

Diaeresis The word ends with a foot end. [4]

The example of a poetic text written in quantitative iamb (see Figure 1.6) might help clarify both definitions. There, a diaeresis can be found, for example, after the words *vidět* and *není* in the second verse. Caesura occurs, for example, after the word *hrozno* in the first verse or the word *milence* in the fourth verse. [3]

Tma | jest | a | hroz|no | vů|kol,
vi|dět | ne|ní | sle|dů;
kte|rá | a|si | ste|zi|čka
ve|de | k mi|len|ce | mé?

Figure 1.6 Caesura and diaeresis (long syllables are underlined)

1.3.2 Metre

The repetition of metrical feet in a verse forms a metre – the abstract outline of a verse. [4]

1.3.3 Rhythm versus metre

The main complexity of the metre assignment task lies in the difference between a rhythm and a metre. When talking about the syllabotonic verse, metre is expressed by the regular alternation of strong and weak positions. On the other hand, rhythm is the poet's actual implementation of the metre using the alternation of accented and non-accented syllables.

For the syllabotonic verse, the underlying concept is that S-positions correspond to accented syllables and V-positions and W-positions to non-accented ones. However, in reality, all positions can correspond to both accented and non-accented ones. In many situations, the poet has the freedom to choose whether to use an accent. As a result, one metre can be expressed by multiple rhythmical patterns.

For the Czech syllabotonic verse, there exist complex rules determining in which situations it is possible to use accented or non-accented syllables. The rules were obtained through a thorough analysis of many poems. Naturally, these rules do not necessarily cover all poems that have ever existed. Sometimes a poem that violates them can be encountered. [4]

1.3.4 Clause (Line ending)

The ending of a verse is called a clause. In the syllabotonic verse, three types of clauses are distinguished based on the last position of a verse:

- masculine,
- feminine,
- acatalectic.

Verses with masculine endings end with the S-position. When the verse ends with W-position, it can either be feminine or acatalectic. The acatalectic verses end with the SVW position pattern, and the feminine verses with the SW pattern. [4] Moreover, as acatalectic are also annotated verses that end with the SV pattern. [6]

1.3.5 Verse multimetry and poem polymetry

A verse is labelled multimetric when its rhythmical pattern can correspond to more metres. The correct metre of such a verse is then selected based on the surrounding context.

A similar concept to multimetry is polymetry, but this time regarding a whole poem. A poem is considered polymetric when some of its verses have different metres assigned than others, and the occurrences of such metres are more or less predictable. [4]

1.3.6 Metrical tagging example

The metrically tagged Czech syllabotonic poetic text (see Figure 1.7) illustrates some of the presented verse properties:

First verse Dactyl with four feet and a masculine clause.

Second verse Dactyl with three feet and an acatalectic clause.

Third verse Dactyl with anacrusis with three feet and a feminine clause.

Fourth verse Dactylotrochee with anacrusis with three feet and a feminine clause. Although accented, the first syllable of the fourth verse represents a weak position. [4]

```
 S  V   W     S  V   W    S  V    W    S
Pra|mé|nek | zaz|vo|nil | ti|še | a | rád. (D4m)
   S  V     W     S   V    W     S  V   W
V srd|ci | mém | poz|dil | se | lis|to|pad (D3a)
W    S  V  W     S   V      W    S  W
a | stu|du|ju | vla|stní | své | ry|sy (Da3f)
W     S  V   W     S  W     S  W
já | za|po|mněl | um|řít | kdy|si. (DTa3f)
```

Figure 1.7 Syllabotonic metrical tagging (Accented syllables are underlined. Strong and weak positions and line tags containing metre, number of feet, and clause are annotated.)

1.4 Special types of verse

In addition to standard metres (see Table 1.1), some special types of verse can also be found in the Czech syllabotonic tradition. Some of them, which are discussed further in the thesis, are presented.

1.4.1 Imitations of hexametre, pentametre, elegiac couplet

Hexametre

Hexametre originally comes from ancient Greek poetry, where it was one of the most widely used metres. Later, it was adopted by the Romans and, from them, spread to medieval Europe. It represents a quantitative dactyl consisting of six feet. An important element of the hexametre verse is a caesura. In the Czech syllabotonic tradition, hexametre imitations began to appear in the 19th century during the Czech National Revival. [4] In the syllabotonic hexametre every line contains 12 to 17 syllables, and its metrical pattern must match the following regular expression (V and W weak positions are annotated with the same symbol):

$$\texttt{\^{}SWW?SWW?SWW?SWW?SWW?SW\$}.\ [5] \tag{1.2}$$

For one of the syllabotonic hexametre poems tagged inside the Corpus of Czech Verse – *Komu platí přízvuk.* by František Vladislav Hek – see Figure 1.8. [6]

S V W S W S V W S V W S V W S W
Sta|teč|ný | A|ga|me|mnon | jak | rych|le | se | do | čes|kých | bás|ní (6f)
S V W S W S W S V W S V W S V
dos|tal, | hned | je|ho | jmě | jsou | na | prs|ty | mě|ři|li. | Jed|ni (6a)
S V W S W S V W S V W S V W S W
vá|ží|ce | ctnost | tak | slav|né|ho | hr|di|ny – | příz|vuč|ně | psa|li; (6f)
S V W S W S W S V W S V W S V
pro | po|hod|lí | však | své, | jmě | na | pra|hu | zkrá|ti|li | dru|zí. (6a)
S W S V W S W S V W S V W S W
Pří|zvuk | pra|vi|dlem | jest – | když | ho|dí | se | na | prs|ty | vol|ně. (6f)

Figure 1.8 Syllabotonic hexametre (Accented syllables are underlined. Strong and weak positions and line tags containing number of feet and clause are annotated.)

Pentametre

The pentametre contains, perhaps surprisingly, not five, but again six dactylic feet. It was rarely used alone; instead, it was used in combination with hexametre inside an elegiac couplet. [4] In the syllabotonic pentametre, every line must contain 10 to 15 syllables, and its metrical pattern must match the pentametre regular expression (V and W weak positions are annotated with the same symbol):

$$\texttt{\^{}SWW?SWW?SW?SWW?SWW?S\$}. \ [5] \qquad (1.3)$$

For an example of a syllabotonic pentametre poem annotated within the Corpus of Czech Verse – *PODZIM V PARKU* by Jaroslav Vrchlický – see Figure 1.9. [6]

S V W S V W S W S W S V W S
Mo|der|ních | či|vů | ty's | v hně|dé | ká|pi | zpo|věd|ník, | vím, (6m)
S V W S W S W S V W S V W S
tlu|mo|číš | věr|ně, | mlč|ky | a | váž|ně: | „Vše|cko | jest | dým!“ (6m)

Figure 1.9 Syllabotonic pentametre (Accented syllables are underlined. Strong and weak positions and line tags containing number of feet and clause are annotated.)

Elegiac couplet

In the syllabotonic elegiac couplet, the hexametres and pentametres alternate regularly. All odd lines correspond to the metrical pattern of the hexameter, and all even lines correspond to the metrical pattern of the pentametre. [5]

1.4.2 Ghazal poems

Ghazals are poems in which the first and every even line contains a so-called *radif* – repeating word or a group of words at the end of the line. The lines containing the radif are then assigned a combination of two different metrical patterns, one pattern for the part without the radif and one pattern for the part containing the radif. Therefore, the resulting metric pattern does not need to correspond to any standard syllabotonic metre. [5]

In the ghazal poem *Vavřín* by Jaroslav Vrchlický (see Figure 1.10), which is annotated within the Corpus of Czech verse, the radif is represented by the word *vavřín.* The part without the radif corresponds to the trochaic metrical pattern. [6] When concatenated with the metrical pattern of the radif part, the resulting metrical pattern does not correspond to any standard syllabotonic metre, as two strong positions next to each other are not allowed (see regular expression (1.1)).

S W S W S W S W S S W
Tma|vé, | smut|né | líst|ky | vy|há|ní | vav|řín, (6f)
S W S W S W S W S S W
hoř|kou | vů|ní | trp|ce | za|vá|ní | vav|řín, (6f)
S W S W S W S W S W S W S
leh|ký, | že | jím | ví|tr | hra|vě | zmí|tá | sem | a | tam, (7m)
S W S W S W S W S S W
bal|va|ny | je | těž|ký | na | skrá|ni | vav|řín. (6f)
S W S W S W S W S W S W
Fir|du|si|ho | hrob | jen | po|sli | ša|cha | naj|dou, (6f)
S W S W S W S W S S W
mi|lu|jeť | vždy | ta|ké | set|ká|ní | vav|řín. (6f)

Figure 1.10 Ghazal poem (Accented syllables are underlined. Strong and weak positions and line tags containing number of feet and clause are annotated.)

Chapter 2

Corpus of Czech Verse

This chapter presents the Corpus of Czech Verse.

The Corpus of Czech Verse is lemmatised, phonetically, morphologically, metrically, rhythmically, and rhyme annotated corpus of Czech poetry from the 19th century and the beginning of the 20th century. [7] It contains 66 428 poems, 2 310 917 lines and 12 636 867 words. [6] It is one of the largest poetic corpora in the world. [7]

2.1 Poem-level annotation

Every poem record stored within the corpus starts with metadata containing information about the poetic book in which the poem was published and the author of the poem. The poem itself is then encoded as a list of lists that divide the poem into stanzas and lines. For a concrete example of the poem-level annotation, see Figure 2.1. [6]

2.2 Line-level annotation

For every line in a poem, the record contains the exact text of the line, a rhyme annotation, and a dictionary that holds all the punctuation. Stress (rhythm) is encoded as a pattern of accented and non-accented syllables. The assigned metres are stored inside a list, allowing for the annotation of multimetric verses. For all possible values of the metrical annotation parameters, see Table 2.1. For an example of the line-level annotation, see Figure 2.2. [6]

At this moment, only syllabotonic verses are metrically annotated. Quantitative, syllabic, and free verses, which also occur in Czech poetry, are classified as "not recognised". [1] However, annotated syllabotonic verses represent the majority of all verses in the corpus – 60 458 (91.01 %) annotated poems, 2 088 508 (90.38 %) annotated lines. [6]

In terms of verse multimetry, 12 182 (0.53 %) lines have more metres assigned. When examining poem polymetry, 2 619 (3.94 %) poems contain more metres. [6]

```
{
    # Metadata on the author of the poem
    'p_author': {
        'born': 1821, # The year author was born
        'died': 1856, # The year author died
        'name': 'Havlíček Borovský, Karel', # Name as printed in the book (it
        ↪  differs from 'identity' in case of a pen name)
        'identity': 'Havlíček Borovský, Karel' # Real name of the author
        },
    # Metadata on book and poem
    'biblio': {
        'motto_aut': None, # Author of the motto
        'b_subtitle': 'Jehly, špičky, sochory a kůly  stesal, zkoval, zostřil,
        ↪  sebral  k vůli  vojně s hloupostí a zlobou místo šavel  Borovský
        ↪  Havel.', # Subtitle of the book
        'publisher': 'Dolenský, Antonín; Unie', # Publisher of the book
        'edition': '[1.]', # Edition description
        'motto': None, # Motto of the book
        'p_title': 'Pražské Vysoké Školy.', # Title of the poem
        'place': 'Praha', # Place where published
        'dedication': None, # Dedication of the book
        'b_title': 'Epigramy', # Title of the book
        'pages': '[80]', # Page range of the poem
        'year': '1921', # Year when published
        'signature': 'ÚČL AV ČR; 52 VIII 2' # Library info
    },
    'book_id': '0176', # ID of the book
    'poem_id': '0001-0004-0000-0001-0000', # ID of the poem
    # Metadata on the author or the editor of the entire book
    'b_author': {
        'born': 1821, # The year author was born
        'died': 1856, # The year author died
        'name': 'Havlíček Borovský, Karel', # Name as printed in the book (it
        ↪  differs from 'identity' in case of a pen name)
        'identity': 'Havlíček Borovský, Karel'}, # Real name of the author
    # The poem itself encoded as a list of lists (stanzas x lines)
    'body': [
        [
            "LINE-LEVEL ANNOTATION",
            ...
        ],
        [...]
    ]
}
```

Figure 2.1 Corpus of Czech Verse: Poem-level annotation

Table 2.1 Corpus of Czech Verse: Metrical annotation parameters values

Parameter	Possible values
Metre	J (iamb)
	T (trochee)
	D (dactyl)
	A (amphibrach)
	X (dactylotrochee)
	Y (dactylotrochee with anacrusis)
	hexameter
	pentameter
	N (not recognized)
Clause	f (feminine)
	m (masculine)
	a (acatalectic)
Foot	Number of feet
Pattern	Pattern of strong (S), weak (W), and undetermined (X) positions

```
{
    # Text of the line
    'text': 'Dvě fakulty v Klementině,',
    # Dict holding punctuation marks
    # Punctuation marks are stored under the key which corresponds to the
    ↪  index of a word which the punctuation precedes
    'punct': {'4': ','},
    # List of words and their metadata
    'words': ["WORD-LEVEL ANNOTATION", ...],
    # Rhyme index (the lines that rhyme all share the same value here)
    'rhyme': 1,
    # List of metres assigned to the line
    'metre': [
        {
            'foot': '4', # Number of feet
            'clause': 'f', # Type of line ending
            'pattern': 'SWSWSWSW', # Pattern of strong and weak positions
            'type': 'T' # Type of metre
        }
    ],
    'stress': '11001000' # Bitstring encoding accented and non-accented
    ↪  syllables
}
```

Figure 2.2 Corpus of Czech Verse: Line-level annotation

2.3 Word-level annotation

On the word level, the corpus provides a lemma (the basic dictionary form), phonetic transcription, and a morphological tag (in the Prague positional tagset format [8]) that contains information on various grammatical categories (part of speech, number, case …). The authors published the phonetic transcription using two formats. [6] The common X-SAMPA [9] format and their own simplified PhoEBE [10] format. For a concrete example of word-level annotation, see Figure 2.3.

```
{
    'token_lc': 'karolínské', # Lowercased token
    'xsampa': 'karoli:nskE:', # X-SAMPA phonetic transcription
    'morph': 'AANS1----1A-----', # Morphological tag
    'phoebe': 'karolInskE', # PHoEBE phonetic transcription
    'token': 'Karolínské', # Word as appears in the text
    'lemma': 'karolínský' # Lemma
}
```

Figure 2.3 Corpus of Czech Verse: Word-level annotation

Chapter 3

Machine learning theory

This chapter aims to familiarise the reader with the machine learning concepts used within this thesis.

3.1 Sequence tagging models

Sequence tagging is an important natural language processing task consisting of receiving a text sequence on input and outputting it tagged. The most famous sequence tagging tasks with defined benchmarks represent, e.g. part of speech (POS) tagging or named entity recognition (NER) tagging, which aims to identify named entities within a text (people, locations, organizations ...). [11]

3.1.1 Recurrent neural network (RNN)

The basic model used for sequence tagging is the recurrent neural network. The RNN model has an input layer x, a hidden layer h and an output layer o. It maintains a "memory" (hidden layer h) containing preliminary information that enables it to predict the output based on previously seen inputs. In the time step t, the following equations are computed:

$$h_t = f(\mathbf{W}_{xh}x_t + \mathbf{W}_{hh}h_{t-1}) \tag{3.1}$$

$$o_t = g(\mathbf{W}_{ho}h_t), \tag{3.2}$$

where

$$f(z) = \frac{1}{1 + e^{-z}} \tag{3.3}$$

$$g(z_m) = \frac{e^{z_m}}{\sum_{k=1}^{m} e^{z_k}}. \tag{3.4}$$

The input to the input layer at time t, x_t, is a vector. The hidden layer updates its value by entering the current input x_t and the previous value of the hidden layer h_{t-1} into the sigmoid activation function f, which returns values between 0 and 1. The recurrent connection between the previous hidden layers and the current hidden layer represents the “memory” of the model. An output at time t, o_t, is a probability distribution over all possible tags. The distribution is obtained from the output of the hidden layer h_t using the softmax activation function g. $\mathbf{W}$ represent weight matrices computed during training time. [11] For a graphical illustration of the network, see Figure 3.1.

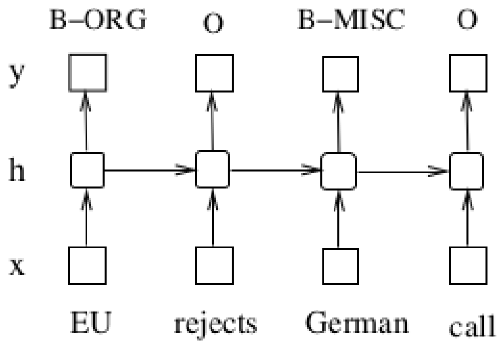

Figure 3.1 RNN [11]

3.1.2 Long short-term memory network (LSTM)

Long short-term memory network represents an upgrade of the RNN network that uses a purpose-built memory cell that contains various gates. The memory cell is represented by an input gate i, a forget gate f, an output gate o, and a cell c. Due to this approach, LSTM might be more successful than RNN in identifying long-range dependencies in data.

In time t the following equations are computed:

$$\begin{aligned} i_t &= \sigma(\mathbf{W}_{xi}x_t + \mathbf{W}_{hi}h_{t-1} + \mathbf{W}_{ci}c_{t-1} + b_i) && (3.5)\\ f_t &= \sigma(\mathbf{W}_{xf}x_t + \mathbf{W}_{hf}h_{t-1} + \mathbf{W}_{cf}c_{t-1} + b_f) && (3.6)\\ c_t &= f_t c_{t-1} + i_t \tanh(\mathbf{W}_{xc}x_t + \mathbf{W}_{hc}h_{t-1} + b_c) && (3.7)\\ o_t &= \sigma(\mathbf{W}_{xo}x_t + \mathbf{W}_{ho}h_{t-1} + \mathbf{W}_{co}c_t + b_o) && (3.8)\\ h_t &= o_t \tanh(c_t). && (3.9) \end{aligned}$$

In the equations, h denotes the hidden vector, $\mathbf{W}$ the weight matrices computed during training, σ the sigmoid activation function, and b the bias. The weight matrices from the cell to the gates (e.g. $\mathbf{W}_{cf}$) are diagonal, and because of this, every element in each gate vector is updated only with the value of the corresponding element in the cell vector. [11]

For an LSTM memory cell schema, see Figure 3.2. For a network schema, see Figure 3.3.

3.1.3 Bidirectional LSTM network (BiLSTM)

Using RNNs and LSTM networks has one major flaw: for each input, the output is predicted using only the previous input information. Information about future inputs cannot be incorporated. The bidirectional LSTM network solves this problem.

The BiLSTM network contains two LSTM networks: forward, which processes the input sequence from left to right, and backward, which processes it from right to left. The output for a time

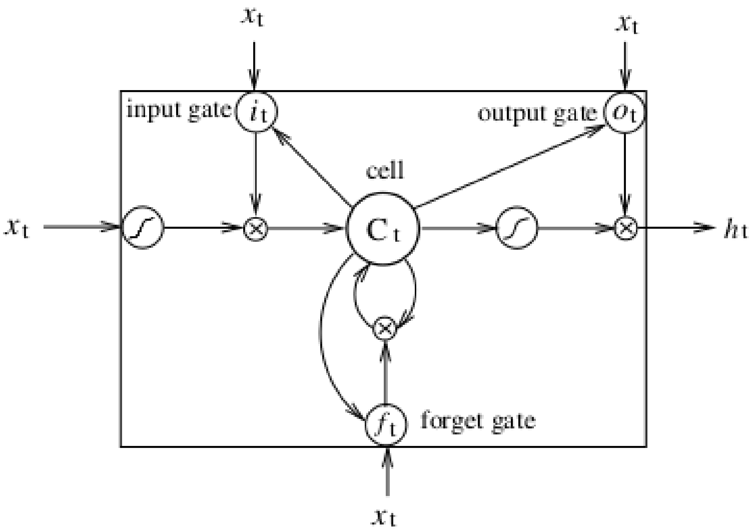

Figure 3.2 LSTM memory cell [11]

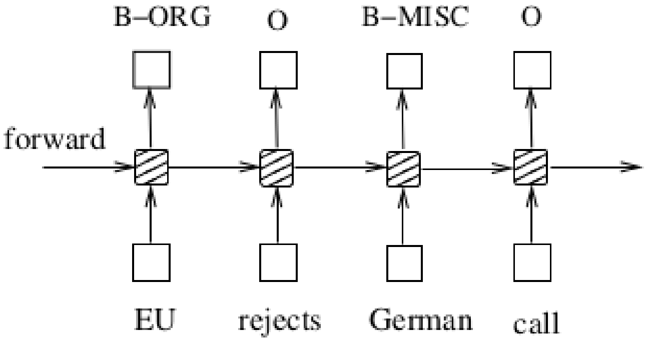

Figure 3.3 LSTM network [11]

step is then obtained using information encoded in both networks. The entire network can be trained using the backpropagation through time algorithm. [11] For a network illustration, see Figure 3.4.

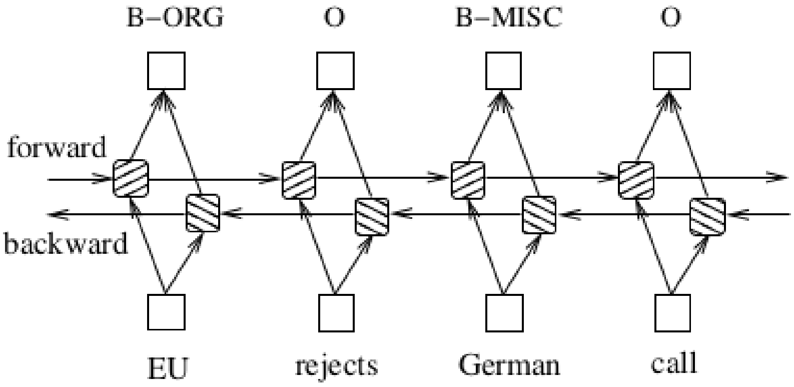

Figure 3.4 BiLSTM network [11]

3.1.4 Conditional random fields network (CRF)

Unlike the previous approaches, conditional random fields focus on the entire sequence rather than individual positions when finding the optimal tagging. In this model, the inputs and outputs are directly connected. There are no recurrent cells as in the previous approaches (see Figure 3.5). [11] For details, see [12].

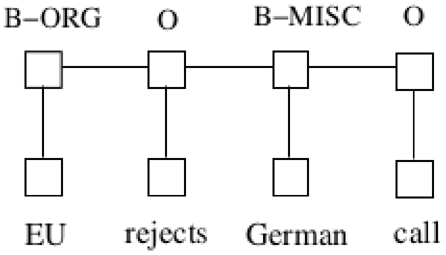

Figure 3.5 CRF network [11]

3.1.5 LSTM-CRF network

LSTM-CRF network combines the LSTM and the CRF network into one model. When predicting the current tag, it can use information on previous inputs from the LSTM layer, as well as sequence-level information on past and future tags from the CRF layer. [11] For an illustration, see Figure 3.6.

3.1.6 Bidirectional LSTM-CRF network (BiLSTM-CRF)

The bidirectional LSTM-CRF network combines the BiLSTM and the CRF network into one model. Unlike the LSTM-CRF network, it uses information about future inputs when predicting

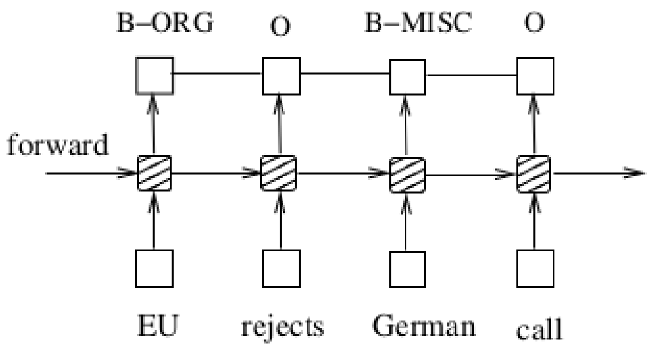

Figure 3.6 LSTM-CRF network [11]

the current tag. It represents the current state-of-the-art model for standard sequence tagging tasks such as POS or NER tagging. [11] For an illustration of the network architecture, see Figure 3.7.

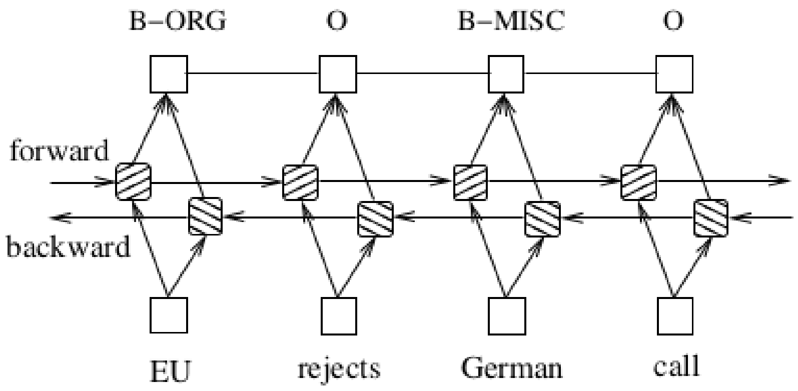

Figure 3.7 BiLSTM-CRF network [11]

Training procedure

The BiLSTM-CRF network is trained using the procedure described in Figure 3.8. Multiple training epochs are performed. In each epoch, the training data are divided into equal-sized batches processed independently. First, the BiLSTM network performs forward passes for both its LSTM networks, forward and backward. As a result, the output score $f_\theta(w)$ is obtained for all tags at all positions, where θ represents the parameters of the BiLSTM network and w represents the input sequence. After that, a CRF layer forward and backward passes are run to compute gradients for the BiLSTM network output and state transition edges in the CRF layer. The errors are then backpropagated to the input using the backward passes of the BiLSTM network. Finally, all network parameters – the state transition matrix of the CRF layer and the parameters θ of the BiLSTM network – are updated. [11]

Input features

Representation of a word inputted into the BiLSTM-CRF network can consist of multiple concatenated vectors: word embedding, capitalisation feature, and character-based representation. [13] For illustration, see Figure 3.9.

```
for each epoch do
   for each batch do
      bidirectional LSTM-CRF model forward pass:
         forward pass for forward state LSTM
         forward pass for backward state LSTM
      CRF layer forward and backward pass
      bidirectional LSTM-CRF model backward pass:
         backward pass for forward state LSTM
         backward pass for backward state LSTM
      update parameters
   end for
end for
```

Figure 3.8 BiLSTM-CRF network: Training procedure

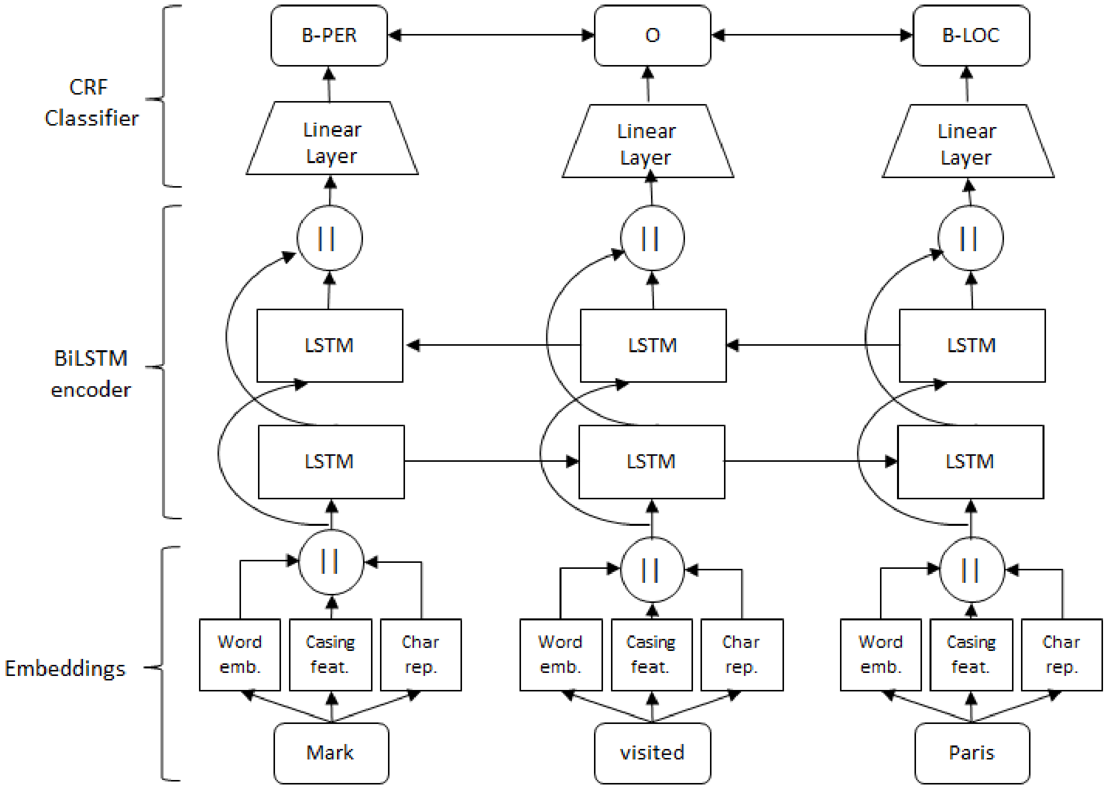

Figure 3.9 BiLSTM-CRF network: Input features [13]

Word embeddings

Word embeddings can be used already pre-trained on large text corpora or can be pre-trained on the training data. Popular algorithms for training word embeddings are, for example, GloVe, FastText (which uses n-grams of words) [13], Word2Vec [14], or ELMo [15].

Capitalisation feature

Since pre-trained word embeddings usually exist only for lowercase words, a one-hot vector representing the original capitalisation tends to be used. It can preserve information about whether the inputted word is entirely or mainly numeric, has all characters upper or lowercase, starts with an uppercase character, or contains some digit. [13]

Character-based representation

Information about the characters of a word can also be added to the input. Two different methods can be used: to derive embedding representing characters of a word using BiLSTM or use convolutional neural networks (CNN). [13]

Training hyperparameters

When training the BiLSTM-CRF, various hyperparameters of the network can be fine-tuned.

Optimizer

While training the network, an objective function is minimised using some optimizer. A commonly used optimizer represents stochastic gradient descent (SGD). However, SGD can be quite sensitive to selecting the learning rate and can have trouble navigating ravines and saddle points. Therefore, other gradient-based optimization algorithms have been proposed, such as Adagrad, Adadelta, RMSProp, Adam, or Nadam (Adam variant that incorporates Nesterov momentum). [13]

Dropout

Dropout is a method that helps prevent overfitting when training a neural network.

Naive dropout The naive dropout strategy applies a randomly selected dropout mask to each LSTM output. Different masks are generated for each time step, and recurrent connections are not dropped. It was noted that this form of dropout is suboptimal for recurrent neural networks.

Variational dropout The optimal dropout for recurrent neural networks represents the variational dropout. It uses the same dropout mask for all time steps. Moreover, it drops the recurrent connections as well. [13]

Gradient clipping and normalization

When a recurrent neural network is trained, two common issues can be encountered: *vanishing gradient* and *exploding gradient* problem. The vanishing gradient problem is countered by using LSTM networks instead of RNNs. To deal with the exploding gradient problem, two common strategies can be applied: gradient clipping and gradient normalization.

Gradient clipping Gradient clipping clips the gradient's components element-wise so that they do not exceed a defined threshold.

Gradient normalization Gradient normalization has a better theoretical justification than gradient clipping. It rescales the gradient whenever the norm of the gradient goes over a defined threshold. [13]

Chapter 4

Metrical analysis of Czech syllabotonic verse

This chapter describes the pipeline to follow when performing the metrical analysis of Czech syllabotonic verse. It presents all subtasks along with possible approaches to solve them.

4.1 Syllabification

When performing a metrical analysis of verse, the first step is to split the text into syllables. Perform the so-called syllabification.

4.1.1 Syllable in Czech

For a native speaker, the question of how many syllables are there in a word is generally relatively easy to answer. Why? Inside every syllable, there is a sonority peak. In Czech, the sonority peak can be expressed by:

1. Vocal or diphthong.
2. Sonorant *r* or *l* when positioned between two consonants or at the end of a word after at least one consonant, e.g. *krk*, *vlk*.
3. In some exceptional cases also nasals or sibilants, e.g. *osm*, *pst*, *sedm*.

Words without the sonority peak – non-syllabic prepositions *v*, *k*, *s*, and *z* – form one syllable with the first syllable of the following word.

On the other hand, determining syllable boundaries is a challenge even for a native speaker. For example, the Czech word *houska*. It obviously consists of two syllables, but how to split it? *Hou-ska* or *hous-ka*? There is no correct answer. [4]

4.1.2 Syllabification using phonetic transcription

So, native speakers can easily determine the number of syllables in a word, but how to do it using a computer? The KVĚTA [5] program applies a sequence of rules to the input words and obtains their phonetic transcriptions. This is possible because the Czech orthography is highly phonemic. The only words that cannot be transcribed using a set of rules are words containing bigrams and foreign words.

As there seems to be no efficient algorithm to automatically decide which instances of the bigrams *au*, *ou* and *eu* represent a diphthong (e.g. *koule*) and which represent two standalone vowels (e.g. *pouliční*), KVĚTA transcribes them using a manually built token-diphthong library and applying a few additional rules. Another problem is irregularities when, for example, words like *nauka* or *Zeus* are treated in some poems as disyllabic and in other poems as monosyllabic. Such irregularities are revealed when KVĚTA tries to assign the most probable metre to a poem. The correct variant must then be selected manually.

When dealing with words from other languages, a possible approach would be to identify the most probable donor language and apply the transcription rules of this language. However, since there are many counterexamples as to why this approach would not be efficient, KVĚTA authors decided to also transcribe foreign words using a manually built library. For some words where the number of syllables differs depending on the case (e.g. *Shakespeare*) or depending on the pronunciation (e.g. *Baudelaire*) the selection of the right variant is also done manually. [5]

4.1.3 Syllabification using hyphenation tools

Thus, for each word, the number of syllables can be obtained from its phonetic transcription. However, is there no way to obtain the number of syllables in a word without having its phonetic transcription? And to also extract the individual syllables? [6]

There exist some tools for the hyphenation of Czech texts. Hyphenation seems like a task similar to syllabification but, in fact, it is something a little different. Hyphenation is nowadays used within every document preparation system (e.g. TEX or any modern web browser) to decide where a word can be split to continue on the following line. There are two approaches to hyphenation:

- etymology-based,
- phonology-based.

Etymology-based systems cut words on the border of a compound word or the border of the stem and ending or prefix or negation. Phonology-based systems cut words based on the pronunciation of syllables. [16] However, they do not cut words directly into individual syllables. Other typographical rules are applied, such as that the first and the last letter of a word cannot be hyphenated (e.g. the word *italština* is hyphenated as *ital-šti-na*) or that words containing less than five letters are not hyphenated. [17]

The approach of using hyphenation tools for syllabification has already been tested by researchers when performing metrical analysis of English and German verse using machine learning. For English verse they decided to train BiLSTM-CRF syllabification model instead, for German verse they used an ensemble of hyphenation tools and heuristic corrections. [18]

4.2 Detecting accented syllables

4.2.1 Accent in Czech

When talking about accent in Czech, usually, each initial syllable of a polysyllabic word is considered accented, while each non-initial non-accented. For monosyllabic words, the rules are not as clear. The general tendency is that the content words (nouns, adjectives, numerals, interjections, and verbs, except for forms of the lemma *být*) are accented, while the function words are non-accented.

A special case is monosyllabic prepositions proper (MPPs), which usually behave as forming a single word with the following one and taking over its accent. However, sometimes MPPs can also be used as standard non-accented function words. This has been largely exploited by poets, especially in the second half of the 19th century. [5] Also the longer the subsequent word, the greater the tendency to have an accent not only on the MPP but also on the subsequent word, e.g. *na mezinárodní letiště*. [4] MPPs are represented by these prepositions: *před*, *od*, *ob*, *ku*, *ke*, *do*, *ve*, *po*, *nad*, *přes*, *při*, *bez*, *se*, *ze*, *za*, *u*, *pod*, *pro* and *zpod*. In the case of two subsequent MPPs, the first is considered a standard function word. [5]

4.3 Metre assignment

In this section, two approaches to the metrical tagging of Czech syllabotonic verse are presented: a rule-based approach that the KVĚTA program used in the past [19] and a data-driven approach that the current version of the KVĚTA program is using. [5]

4.3.1 Rule-based KVĚTA algorithm

All the following information is taken from [19].

Prosodic analysis

The first step of the rule-based algorithm is to represent a line of verse as a string containing symbols *0*, *1* and *+*, where *1* represents an accented syllable, *0* a non-accented syllable, and a *+* a monosyllable. Monosyllables can be both accented and non-accented. In the case of a monosyllabic preposition, the preposition itself is treated as an initial syllable of a polysyllabic unit and the following syllable as a non-initial syllable of a polysyllabic unit. For the transcription rules, see Table 4.1.

Table 4.1 Rule-based KVĚTA algorithm: Prosodic transcription rules

Syllable type	Symbol
First syllable of a polysyllabic unit	1
Non-initial syllable of a polysyllabic unit	0
Monosyllable	+

The sentence “Za trochu lásky šel bych světa kraj” is represented by a primary string:

`10010++10+.`

The problem is that poets do not produce purely metrical lines. Therefore, in addition to the primary string, the algorithm also analyses three alternative strings applying the three most common stress alterations:

1. Treating the sequence of a monosyllabic preposition and another unit as having the stress located not on the first syllable (preposition) but on the second one (`100 -> +10`).
2. Relocating the stress of a polysyllabic unit (incidental prepositions included) to the immediately preceding monosyllable (`+10 -> 100`).
3. Tolerating that stress of one polysyllabic unit occupies one of the forbidden positions.

Metrical analysis

The metrical analysis is based on the following simplified rules of Czech syllabotonic verse:

1. A line is iambic (I) if no odd position except the first (allowing dactylic incipits) is occupied by the stress of a polysyllabic unit.
2. A line is trochaic (T) if no even position is occupied by the stress of a polysyllabic unit.
3. A line is dactylic (D) if for each $n = 0, 1, 2, \ldots$
   a. no (3n+3)$^{\text{rd}}$ position is occupied by the stress of a polysyllabic unit and
   b. no (3n+2)$^{\text{nd}}$ position is occupied by the stress of a unit consisting of three or more syllables and
   c. every stress occupying a (3n+2)$^{\text{nd}}$ position is preceded by a monosyllabic unit.
4. A line is dactylic with anacrusis (Da) if for each $n = 0, 1, 2, \ldots$
   a. the first position is occupied by a monosyllabic unit and
   b. no (3n+4)$^{\text{th}}$ position is occupied by the stress of a polysyllabic unit and
   c. no (3n+3)$^{\text{rd}}$ position is occupied by the stress of a unit consisting of three or more syllables and
   d. every stress occupying a (3n+3)$^{\text{rd}}$ position is preceded by a monosyllabic unit.
5. A line is dactylo-trochaic (DT) if a “virtual syllable” can be inserted into the line after some of the 3n+2 nd positions (at least once) in order to meet the conditions specified in 3.
6. A line is dactylo-trochaic with anacrusis (DTa) if the “virtual syllable” can be inserted into the line after some of the 3n+3$^{\text{rd}}$ positions (at least once) in order to meet the conditions specified in 4.

The metrical tagging procedure is visualised on a sample from Karel Hynek Mácha’s *Máj* – an iambic poem, yet with iambic constraints frequently violated in various ways. Row 0 shows the primary string, and rows I-III show the alternative strings that allow for stress alterations. Positions within the string that violate the constraints of a given metre are highlighted. The first string in a column that does not violate the constraints of a given metre or violates it only once (row III) is underlined:

Zhasla měsíce světlá moc,

Row	Iamb	Trochee	Dactyl	Dactyl with anacrusis
0	10**1**0010+	10100**1**0+	10**1**00**1**0+	**1**0**1**00**1**0+
I	10**1**0010+	10100**1**0+	10**1**00**1**0+	**1**0**1**00**1**0+
II	10**1**0010+	10100**1**0+	10**1**00**1**0+	**1**0**1**00**1**0+
III	10**1**0010+	10100**1**0+	10**1**00**1**0+	**1**0**1**00**1**0+

Output: Iamb / Trochee

i hvězdný svit a kol a kol

Row	Iamb	Trochee	Dactyl	Dactyl with anacrusis
0	+10+++++	+**1**0+++++	+10+++++	+10+++++
I	+10+++++	+**1**0+++++	+10+++++	+10+++++
II	100+++++	100+++++	100+++++	**1**00+++++
III	+10+++++	+**1**0+++++	+10+++++	+10+++++

Output: Iamb / Trochee / Dactyl / Dactyl with anacrusis

je pouhé temno, širý dol

Row	Iamb	Trochee	Dactyl	Dactyl with anacrusis
0	+101010+	+**1**0**1**0**1**0+	+1010**1**0+	+10**1**0**1**0+
I	+101010+	+**1**0**1**0**1**0+	+1010**1**0+	+10**1**0**1**0+
II	1001010+	100**1**0**1**0+	10010**1**0+	**1**00**1**0**1**0+
III	+101010+	+**1**0**1**0**1**0+	+1010**1**0+	+10**1**0**1**0+

Output: Iamb / Dactyl

co hrob daleký zívá.

Row	Iamb	Trochee	Dactyl	Dactyl with anacrusis
0	++**1**0010	++100**1**0	++**1**00**1**0	++**1**00**1**0
I	++**1**0010	++100**1**0	++**1**00**1**0	++**1**00**1**0
II	+100010	+**1**000**1**0	+**1**000**1**0	+1000**1**0
III	++**1**0010	++100**1**0	++**1**00**1**0	++**1**00**1**0

Output: Iamb / Trochee

The final metre for the poem is selected based on a "metrical index" value. The metrical index of a line is computed as follows:

- If at any step a line is assigned a metre, the metrical index of that metre has the value of 100.
- Every stress of a disyllabic unit according to rules 3b/3c or 4b/4c except for the second position (allowing iambic incipits in the dactylic verse) lowers the metrical index value by 20.
- If the metre is assigned at row I, metrical index is lowered by 40.
- If the metre is assigned at row II, metrical index is lowered by 60.
- If the metre is assigned at row III, metrical index is lowered by 80.
- Negative values are set to 0.

For every metre, the arithmetic mean of the line metrical indices is computed. The metre with the highest mean value, that further meets the condition of having the mean value higher than

50 and containing no line with a metrical index of 0, is finally selected. For the Mácha's *Máj* example, Iamb is selected:

	Iamb	Trochee	Dactyl	Dactyl with anacrusis
Zhasla měsíce světlá moc,	20	20	0	0
i hvězdný svit a kol a kol	100	40	100	100
je pouhé temno, širý dol	100	0	20	0
co hrob daleký zívá.	40	20	0	0
Arithmetic mean	65	20	30	25

Multimetric lines

In this approach, the metre for a multimetric line is selected depending on the context: the metres of other lines. For multimetric poems that can be assigned multiple metres to an equal degree, preference rules are applied.

Conclusion

To conclude, this algorithm can be used for the metrical tagging of Czech syllabotonic verse. Because of the inability of the algorithm to decide whether a monosyllable is accented or not, distinctions among some rare types of dactylo-trochee may not be recognised. Polymetric poem tagging is also not possible. Furthermore, this algorithm does not tag hexametre and pentametre poems, which are tagged within the Corpus of Czech Verse.

4.3.2 Data-driven KVĚTA algorithm

All the following information comes from [5].

Syllable classes

Table 4.2 Data-driven KVĚTA algorithm: Syllable class structure

Parameter	Values
initial	1 (word-initial syllable) 0 (non-initial syllable)
final	1 (word-final syllable) 0 (non-final syllable)
contentWord	1 (content-word) 0 (function word)
preposition	1 (MPP preposition) 0 (other word)
prevPreposition	1 (preceded by MPP preposition) 0 (preceded by other word)
prevInitial	1 (preceded by word-initial syllable) 0 (preceded by non-initial syllable)
nextlong	1 (followed by syllable containing long vowel or diphtong) 0 (followed by other syllable)

The data-driven KVĚTA approach represents every syllable as a so-called Syllable class. The Syllable class is a data class containing Boolean parameters (see Table 4.2) extracted from the Czech accent rules. A poem is then internally represented as a list of lists of Syllable classes, the division into stanzas is not taken into account.

In theory, there are 128 different combinations of parameter values, but in practise many of those are discarded as, e.g. `{initial: 0, preposition: 1}` or `{final: 0, preposition: 1}`. Another example is the contentWord parameter, which is used only for monosyllabic words that are not preceded by MPP preposition `{initial: 1, final: 1, prevPreposition: 0}`. In addition, the parameters nextLong and prevInitial are taken into account only for a word-initial syllable of a polysyllabic word `{initial: 1, final: 0}` or MPP preposition `{preposition: 1}`. More such reductions are applied, resulting in only 12 recognised Syllable class instances.

For example, the Syllable class instance for the third syllable (*Zná*) of a line: “Aj! kdo zná ji, tu osobu” is:

```
{
    initial: 1,
    final: 1,
    contentWord: 1,
    preposition: 0,
    prevPreposition: 0,
    prevInitial: 1,
    nextLong: 0
}.
```

Metre generation

The next step of the algorithm is to take for each poem only the number of syllables in each line and generate all possible metres that could fit it. Not only standard syllabotonic metres are generated, but also syllabotonic imitations of quantitative syllabic strophes and syllabotonic imitations of the quantitative hexametre, pentametre, and elegiac couplet. Ghazal poems are generated as well. The V-positions are not distinguished and are represented as W-positions.

Standard syllabotonic metres

The maximal number of syllables in one line `max_syll` must be determined when generating all standard syllabotonic metres for a poem. All possible metres are then generated as strings of length `max_syll` that match the regular expression (1.1).

This regular expression represents all valid combinations of Czech syllabotonic feet. For example, if `max_syll` for a poem is equal to 6, the following metres are generated:

```
[‘SWSWSW’, ‘SWWSWS’, ‘SWSWWS’, ‘SWWSWW’, ‘WSWSWS’, ‘WSWSWW’, ‘WSWWSW’].
```

For every metre generated, a two-dimensional array of metrical positions is created, where the metrical pattern of each line is a prefix of the generated metre. For metre `'SWWSWS'` and a poem consisting of a 5-syllable line, a 6-syllable line, and a 3-syllable line, the following array is generated:

```
metre['SWWSWS'] = [
    ['S', 'W', 'W', 'S', 'W'],
    ['S', 'W', 'W', 'S', 'W', 'S'],
    ['S', 'W', 'W']
].
```

Imitations of quantitative syllabic strophes

The algorithm supports the four most commonly imitated quantitative syllabic strophes: the Sapphic Stanza, the Third Asclepiad Stanza and two types of the Alcaic Strophe. When the poem meets the syllabic requirements and can be divided into one of these strophes, the algorithm includes this option in the generated metres. For the metrical patterns of the strophes, see Figure 4.1.

```
metre['sapphic'] = [
    ['S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W'],
    ['S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W'],
    ['S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W'],
    ['S', 'W', 'W', 'S', 'W']
]

metre['asclepiad'] = [
    ['S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'W'],
    ['S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'W'],
    ['S', 'W', 'S', 'W', 'W', 'S', 'W'],
    ['S', 'W', 'S', 'W', 'W', 'S', 'W', 'W']
]

metre['alcaicA'] = [
    ['W', 'S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'W'],
    ['W', 'S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'W'],
    ['W', 'S', 'W', 'S', 'W', 'S', 'W', 'S', 'W'],
    ['S', 'W', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W']
]

metre['alcaicB'] = [
    ['S', 'W', 'S', 'W', 'W', 'S', 'W', 'W', 'S', 'W', 'W'],
    ['S', 'W', 'S', 'W', 'W', 'S', 'W', 'W', 'S', 'W', 'W'],
    ['S', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'W'],
    ['S', 'W', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W']
]
```

Figure 4.1 Syllabotonic imitations of quantitative syllabic strophes: Metrical patterns

Imitations of hexametre, pentametre and elegiac couplet

For a poem that imitates the quantitative hexametre, every line must match the hexametre regular expression (1.2). That implies that the length of every line varies from 12 to 17 syllables. Moreover, when the length of the line lies between 13 and 16 syllables, there is more than one possible metrical pattern. There are five different patterns for lines of lengths 13 and 16 and ten patterns for lines of lengths 14 and 15. Therefore, a three-dimensional array instead of a two-dimensional array must be generated. For a poem consisting of a 13-syllable line and a 12-syllable line, the following array is generated:

```
metre['hexametre'] = [
    [
        ['S', 'W', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'S', 'W'],
        ['S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'S', 'W'],
        ['S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W', 'S', 'W'],
        ['S', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W', 'S', 'W'],
        ['S', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'W', 'S', 'W']
    ],
    [
        ['S', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'S', 'W', 'S', 'W']
    ]
].
```

For imitation of the quantitative pentametre, the situation is analogous. This time, all lines must match the pentametre regular expression (1.3). Line lengths range from 10 to 15 syllables, again resulting in a three-dimensional array.

In imitation of the elegiac couplet, all odd lines must match the hexametre regular expression, and all even lines must match the pentametre regular expression. Again, a three-dimensional array is returned.

Ghazal poems

When a poem is identified as a ghazal poem, the algorithm splits the poem into two separate parts – part without the radifs and a part containing only the radifs. Possible metrical patterns are generated for each part separately, and the final metrical patterns are returned as their Cartesian product.

Metre assignment

Metrical coefficient

To be able to assign metres, the algorithm must pre-calculate the values of “metrical coefficient” from the corpus data. For every Syllable class, the metrical coefficient value is represented as the conditional probability that the Syllable class is realised by a strong or weak position in the corresponding metrical pattern. For the j^{th} syllable of the i^{th} line of a poem, and a metre w (which was generated for this poem as a possible metre in the previous step of the algorithm) let $\mu_{w,i,j}$ be the value of the metrical coefficient for that syllable and metre w, $\sigma_{i,j}$ the Syllable class assigned to that syllable and $x_{w,i,j}$ the strong or weak position in the metrical pattern of metre w.

The authors of the paper use Bayes’ Theorem (under a naive independence assumption):

$$\mu_{w,i,j} = P(x_{w,i,j}|\sigma_{i,j}) = \frac{P(\sigma_{i,j}|x_{w,i,j}) \cdot P(x_{w,i,j})}{P(\sigma_{i,j}|S) + P(\sigma_{i,j}|W)} \tag{4.1}$$

and argue that since $x_{w,i,j}$ is a binary value with possible values S or W, the probability of which may be considered equal, they can internally compute $\mu_{w,i,j}$ just as:

$$\tilde{\mu}_{w,i,j} = \frac{P(\sigma_{i,j}|x_{w,i,j})}{P(\sigma_{i,j}|S) + P(\sigma_{i,j}|W)}. \tag{4.2}$$

Moreover, the algorithm also takes into account that the probabilities can vary greatly depending on the time period or individual authors. Therefore, for each author, the Yates χ^2 test is performed to determine whether the probabilities obtained from his work differ statistically significantly from the probabilities of the entire corpus. If they do, his values of the metrical index are set as a geometric mean of the probabilities from the whole corpus and of his probabilities (P_A) with a weight ratio of 1 to 3:

$$\mu_{w,i,j} = \sqrt[4]{P(x_{w,i,j}|\sigma_{i,j}) \cdot (P_A(x_{w,i,j}|\sigma_{i,j}))^3}. \tag{4.3}$$

Metre selection

When selecting a metre, the metrical coefficients of the individual line patterns and the overall metrical coefficient must be determined for every generated metre.

For the i^{th} line of a poem and a metre w, the metrical coefficient of the line pattern $L_{w,i}$ represents the probability of this line being realised by the metre w. It is computed as a product of metrical coefficients of all individual syllables on the line, and it is normalised by the length of the line denoted as n_i, so lines of different lengths are comparable:

$$L_{w,i} = \sqrt[n_i]{\prod_{j=1}^{n_i} \tilde{\mu}_{w,i,j}}. \tag{4.4}$$

In the case of the three-dimensional hexametre, pentametre, or elegiac couplet arrays, the line pattern with the highest value of $L_{w,i}$ is selected (out of k possible patterns):

$$L_{w,i} = \max\{L_{w,i,1}, L_{w,i,2}, \ldots, L_{w,i,k}\}. \tag{4.5}$$

The overall metrical coefficient T_w of the metre w is calculated as a geometric mean of the metrical coefficients of all line patterns:

$$T_w = \sqrt[m]{\prod_{i=1}^{m} L_{w,i}}, \tag{4.6}$$

where m is the number of lines.

Finally, the metre w with the highest metrical coefficient T_w is selected. Inside KVĚTA, various checks must be passed to annotate the poem automatically. For example, if the value of T_w or some $L_{w,i}$ is too low, or the standard deviation of $L_{w,1}, L_{w,2}, \ldots$ is too high, manual control is required. This approach minimises the incorrect annotations of polymetric or non-syllabotonic poems and reveals possible mistakes in phonetic transcriptions. When the same metrical pattern is generated with different metre names (e.g. hexametre and trochee), standard syllabotonic metres are preferred.

Conclusion

In conclusion, this algorithm seems to be more suitable for metrical tagging of Czech syllabotonic verse than the rule-based one. It can distinguish whether a monosyllable is accented or not, it allows for more advanced rules as e.g. the MPPs are also taken into account, and it also tags hexametres and pentametres, which are tagged within the Corpus of Czech Verse. Again, polymetric tagging of poems is not possible.

4.4 Machine learning approaches

The metrical analysis can be modelled as a sequence tagging task and solved using machine learning. In the paper [18], conditional random fields, BERT model and BiLSTM-CRF model are tested on English and German verse corpora. The paper [2] tests perceptron, hidden Markov model (HMM), conditional random fields, and BiLSTM-CRF on English and Spanish verse corpora. Both articles evaluate syllable-level and line-level accuracies and obtain the best results using BiLSTM-CRF.

4.4.1 BERT

The authors of [18] assume that BERT's transformer architecture cannot compete against other models, possibly because of an improper syllable representation, as BERT's word-piece tokenizer creates word chunks that are not equivalent to syllables.

Another experiment is performed in which BERT should learn the verse label (e.g. iambic pentameter) for a given sequence of word tokens. BERT detects frequent classes like the iambic pentameter or the trochaic tetrameter well but fails to learn measures other than iamb, trochee and irregular verse with inversions. The authors assume that BERT probably mainly predicts based on the length of lines in this setting.

4.4.2 BiLSTM-CRF

In [18], individual syllables are sent to the input and custom syllable embeddings are pre-trained from verse corpora using the Word2Vec algorithm. BiLSTM-CRF uses three BiLSTM layers with 100 recurrent units in each layer and uses a linear-chain CRF classifier. Variational dropout is used, with 25 % dropped in output and recurrent connections. No character-based representation of syllables is used, as it, according to the authors, hurts both speed and accuracy.

Unlike the intuition denoted in [18], in [2], BiLSTM-CRF is used with BiLSTM character-based representation of input tokens. Three different input types are tested: individual syllables, word tokens, and individual syllables with word boundaries. For English, the best results are obtained for individual syllables with word boundaries in the input. For Spanish, the best results are obtained when word tokens are inputted.

Chapter 5

Implementation

This chapter discusses approaches aiming to solve the metrical analysis task. First, the reimplementation of the KVĚTA data-driven approach, later using machine learning and training the BiLSTM-CRF model.

5.1 Used datasets

Two different datasets obtained from the Corpus of Czech Verse data are used.

The first dataset contains no polymetric poems, no multimetric verses, no unrecognised metrical positions, and no annotation errors. This dataset contains 57 339 poems.

The second dataset contains no unrecognised metrical positions, no multimetric verses, and no annotations errors as well. However, this time, polymetric poems can be present. This dataset contains 59 661 poems.

Both datasets are divided into training, validation, and testing data using a ratio of 70:15:15. The KVĚTA reimplementation is trained on the training and validation data and tested on the testing data. The BiLSTM-CRF model is trained on the training data, finetuned during training on the validation data, and final results are obtained for the previously unseen testing data.

5.2 KVĚTA reimplementation

5.2.1 Syllabification using X-SAMPA transcription

In the reimplementation of the KVĚTA data-driven approach, a phonetic transcription algorithm is not implemented. Instead, phonetic transcriptions already generated by KVĚTA and published within the Corpus of Czech Verse are used. The X-SAMPA format is parsed to obtain the number of sonority peaks and, therefore, the number of syllables for each word. For the X-SAMPA symbols that are counted, see Tables 5.1, 5.2, 5.3, and 5.4.

Table 5.1 X-SAMPA: diphthongs

X-SAMPA Transcription	Example Czech Word	Word Transcription
a_u	auto (car)	a_uto / ?a_uto
E_u	neuron (neuron)	nE_uron
o_u	soud (court)	so_ut

Table 5.2 X-SAMPA: vocals

X-SAMPA Transcription	Example Czech Word	Word Transcription
a	sad (orchard)	sat
E	pes (dog)	pEs
I	list (leaf)	lIst
o	rok (year)	rok
u	sud (barrel)	sut

Table 5.3 X-SAMPA: long vocals

X-SAMPA Transcription	Example Czech Word	Word Transcription
a:	sám (alone)	sa:m
E:	lék (medicine)	lE:k
i:	sýr (cheese)	si:r
o:	tón (tone)	to:n
u:	sůl (salt)	su:l

Table 5.4 X-SAMPA: syllabic consonants

X-SAMPA Transcription	Example Czech Word	Word Transcription
=	krk, vlk (throat, wolf)	kr=k, vl=k

5.2.2 Syllable classes

In the paper [5] explaining the KVĚTA data-driven approach, the final Syllable class count is reduced to only 12 different Syllable classes. As not all the reductions applied are properly explained in the paper, the reimplementation got to the number of 15 different Syllable classes.

5.2.3 Metre generation

The reimplementation generates the same metres as KVĚTA – the standard syllabotonic metres; imitations of quantitative syllabic strophes; imitations of hexametre, pentametre, and elegiac couplet; and ghazal poems.

5.2.4 Metre assignment

In the reimplementation, the conditional probability of a Syllable class being realised by a strong or weak position $P(x_{w,i,j}|\sigma_{i,j})$ is not counted using Bayes' Theorem as it is inside KVĚTA (see Equation (4.2)), but is counted directly. In my opinion, counting the probability using Bayes' theorem is, in this case, unnecessary and does not add any benefit compared to counting it directly. Moreover, in my opinion, the probabilities of strong and weak positions should not be considered equal, as the weak position is a bit more likely to occur depending on the structure of all the syllabotonic feet. Probabilities are estimated from training and validation datasets. Yates' χ^2 test is not performed as it adds another complexity to the task, and, therefore, the probabilities are not modified for outlier authors (see Equation (4.3)).

When selecting the metre, the metrical coefficients of the individual line patterns and the overall metrical coefficients are calculated the same as in KVĚTA (see Equations (4.4), (4.5), and (4.6)). When the same metrical pattern is generated with different metre names, standard syllabotonic metres are selected with priority as in KVĚTA.

5.3 BiLSTM-CRF

Based on the research of machine learning approaches done in Section 4.4, the BiLSTM-CRF model is selected for training.

5.3.1 Used implementation

One of the publicly available implementations of BiLSTM-CRF for sequence tagging [20] is used to train the BiLSTM-CRF model. The same implementation is used in the paper [18]. It uses Keras version 2.2.0 with Tensorflow 1.8.0 as the backend. It must be run using Python 3.6 or lower to be compatible with the Tensorflow version. [20]

5.3.2 Input data format

When training the BiLSTM-CRF model, two different input data formats are tested:

- token-level,

- syllable-level.

Both formats are inputted using two different approaches:

- one input sequence represents a line in a poem,
- one input sequence represents a whole poem.

For the token-level input data format, the model is inputted a sequence of individual tokens (words) from the poem. For each token, a sequence of metrical positions is predicted. This sequence can be empty as well, when the token is non-accented.

For the syllable-level input data format, the model is inputted a sequence of individual syllables from the poem. For every syllable, exactly one metrical position is predicted. Because the Corpus of Czech Verse lacks annotation of the syllable boundaries, two approaches to obtain syllable-level data are tested: syllabification using hyphenation tools and X-SAMPA syllables.

Syllabification using hyphenation tools

Approach to syllabification using phonology-based Czech TEX hyphenation patterns [21] with Frank Liang's hyphenation algorithm [22] is tested. The implementation of Frank Liang's algorithm published by Ned Batchelder [17] is modified to support Czech patterns. The problem encountered is that, for example, the word *houska* is hyphenated as *hou-s-ka* due to the ambiguity of the syllable boundaries where both *hous-ka* and *hou-ska* represent valid syllable splits. Therefore, the output has to be further processed to identify and merge these splits.

The next hyphenation tool tested is the Pyphen [23] program, which uses Hunspell hyphenation dictionaries and supports the Czech language. It does not annotate both syllable boundaries in case of ambiguity, as Czech TEX hyphenation patterns do.

For the training data of the second dataset used (see Section 5.1) the average difference from the correct number of syllables in one token is 0.21 for Czech TEX hyphenation patterns, 0.06 for Pyphen and 0.15 for Czech TEX hyphenation patterns with applied heuristics, that one-letter syllables are merged into the previous ones. For Czech TEX hyphenation patterns, on average, 21.63 % of tokens in one line have an incorrect number of syllables assigned, for Pyphen 6.18 % and for Czech TEX hyphenation patterns with heuristics 15.31 %. The referential syllable counts are obtained by splitting X-SAMPA transcriptions using sonority peak symbols (see Tables 5.1, 5.2, 5.3, and 5.4).

As the differences between the referential syllable counts and the syllable counts obtained by the hyphenation approaches are not so insignificant, it is decided not to use the hyphenation tools at the moment. Furthermore, knowledge about non-syllabic prepositions would have to be incorporated when using these approaches.

X-SAMPA syllables

As the hyphenation approaches to syllabification are rejected, X-SAMPA phonetic transcriptions are parsed to obtain syllable representations with correct syllable count. For each word, its X-SAMPA transcription is divided into X-SAMPA syllables using transcriptions of sonority peaks (see Tables 5.1, 5.2, 5.3, or and 5.4). For non-syllabic words without a sonority peak, no X-SAMPA syllables are generated.

As an example, for the token *blýskajícím* with X-SAMPA transcription `bli:skaji:t_si:m` the following four X-SAMPA syllables are obtained: `[bli:, ska, ji:, t_s:m]`.

5.3.3 Input features

In accordance with the intuition of the paper [18] no character-based representation of the input is used. The decision is made not to use the capitalisation feature, as the capitalisation of words seems to be of minor importance for this task, and words inside poems usually do not contain numerals.

As word embeddings, Word2Vec embeddings are pre-trained on the training data. By mistake, for the syllable input, Word2Vec embeddings are pre-trained also on the non-syllabic tokens that are not inputted to the model. Due to time reasons, the models are not retrained. However, the assumption is that it should not affect the results significantly.

The implementation supports inputting other features in addition to the input tokens. Following intuition from [5], author of a poem, year of publication, POS tag, or lemma is also inputted. An embedding of size 10 is assigned automatically to every input value and concatenated with other input embeddings.

5.3.4 Training

Training loop denoted in Figure 3.8 is performed.

The implementation is optimised for speed, it groups sentences with the same lengths together during training, and thanks to that, it is much faster than other BiLSTM-CRF implementations. [20] Even though the implementation is highly configurable, the hyperparameters are not further fine-tuned, as training of one input configuration takes 1 to 2 days (CPU is used, training on GPU is not tested).

5.3.5 Training hyperparameters

The hyperparameters are set according to the recommendations in [18] and [13].

Following intuition from [18], variational dropout is used, which drops 25 % in the output and recurrent connections, and three BiLSTM layers with 100 recurrent units are trained inside each layer.

In accordance with the findings of [13], Nadam optimizer is utilised. Thirty-two sentences are used for mini-batch training, as the training set is relatively large. To fight the exploding gradient problem, no gradient clipping, but gradient normalization is performed with threshold value 1. Models are trained for 15 epochs. Training is stopped earlier if the best validation score does not increase for more than five training epochs.

5.3.6 Example configuration of hyperparameters

For a concrete example of the configuration of hyperparameters, see Figure 5.1.

```
{
    "dropout": [0.25, 0.25],
    "classifier": ["CRF"],
    "LSTM-Size": [100, 100, 100],
    "optimizer": "nadam",
    "earlyStopping": 5,
    "miniBatchSize": 32,
    "charEmbeddings": None,
    "clipvalue": 0,
    "clipnorm": 1,
    "featureNames": ["tokens", "lineIdx", "author", "year", "pos", "lemma"],
    "addFeatureDimensions": 10
}
```

Figure 5.1 BiLSTM-CRF library: Example configuration of hyperparameters

Chapter 6

Results

6.1 Accuracies

When testing the metrical analysis approaches implemented, the following accuracies are taken into account:

Syllable-level accuracy : Testing for all individual syllables whether the metrical position assigned to that syllable is the same as the referential.

Line-level accuracy : Testing for all individual lines whether the metrical pattern assigned to that line is the same as the referential.

Poem-level accuracy : Testing for all individual poems whether the metrical pattern assigned to that poem is the same as the referential.

For BiLSTM-CRF models with token input, syllable accuracies cannot be evaluated because the reference syllable counts and predicted syllable counts tend to differ.

6.2 Results

For the results on both datasets used (see Section 5.1), see Tables 6.1, 6.2.

It can be seen that BiLSTM-CRF represents a great success, as with the best input configurations, it returns better results than the KVĚTA reimplementation, comparing all evaluated accuracies (syllable-level, line-level, and poem-level).

Not surprisingly, the syllable input returns better results than the token input, especially when comparing the poem-level accuracy. However, when only the line-level accuracy is looked at, the results for token input are not that bad. Therefore, these results seem promising; as for token input, prior syllabification is not needed during preprocessing. An interesting observation is that inputting a lemma significantly improves the results for token input. Generally, the more additional features inputted (author of a poem, year of publication, POS tag, lemma), the better the results.

However, the most interesting finding of this work is how much inputting sequences representing whole poems rather than poem lines improves the results (especially the poem-level accuracy and perhaps surprisingly even the syllable-level accuracy). The best results are obtained inputting sequences representing whole poems with all additional features and poem line indices on the input. Poem line indices allow the model to distinguish different lines of a poem.

Table 6.1 Results for the first dataset (no polymetric verses, no verse multimetry)

Approach	Syll. acc.	Line acc.	Poem acc.
KVĚTA reimplementation	97.02	81.88	69.83
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables Max epochs: 15	97.93	95.67	66.78
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables, authors, years Max epochs: 15	98.14	95.97	66.29
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables, authors, years, POS tags Max epochs: 15	98.69	96.85	72.34
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables, authors, years, POS tags, lemmas Max epochs: 15	98.74	96.94	72.98
BiLSTM-CRF with X-SAMPA syll. embeddings poem as line Input: line idxs, X-SAMPA syllables, authors, years, POS tags, lemmas Max epochs: 15	**99.71**	**99.17**	**92.51**
BiLSTM-CRF with tokens embeddings Input: tokens Max epochs: 15	-	74.38	8.69
BiLSTM-CRF with tokens embeddings Input: tokens, authors, years Max epochs: 15	-	75.91	10.29
BiLSTM-CRF with tokens embeddings Input: tokens, authors, years, POS tags Max epochs: 15	-	76.94	11.17
BiLSTM-CRF with tokens embeddings Input: tokens, authors, years, POS tags, lemmas Max epochs: 15	-	90.01	31.83
BiLSTM-CRF with tokens embeddings poem as line Input: line idxs, tokens, authors, years, POS tags, lemmas Max epochs: 15	-	95.47	55.74

Table 6.2 Results for the second dataset (polymetric verses, no verse multimetry)

Approach	Syll. acc.	Line acc.	Poem acc.
KVĚTA reimplementation	96.08	80.64	68.26
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables Max epochs: 15	97.78	95.31	62.91
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables, authors, years Max epochs: 15	98.21	96.09	67.20
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables, authors, years, POS tags Max epochs: 15	98.72	96.96	73.41
BiLSTM-CRF with X-SAMPA syll. embeddings Input: X-SAMPA syllables, authors, years, POS tags, lemmas Max epochs: 15	98.83	97.12	73.06
BiLSTM-CRF with X-SAMPA syll. embeddings poem as line Input: line idxs, X-SAMPA syllables, authors, years, POS tags, lemmas Max epochs: 15	**99.61**	**98.86**	**90.40**
BiLSTM-CRF with tokens embeddings Input: tokens Max epochs: 15	-	72.14	7.56
BiLSTM-CRF with tokens embeddings Input: tokens, authors, years Max epochs: 15	-	76.60	9.64
BiLSTM-CRF with tokens embeddings Input: tokens, authors, years, POS tags Max epochs: 15	-	78.40	11.27
BiLSTM-CRF with tokens embeddings Input: tokens, authors, years, POS tags, lemmas Max epochs: 15	-	91.42	35.39
BiLSTM-CRF with tokens embeddings poem as line Input: line idxs, tokens, authors, years, POS tags, lemmas Max epochs: 15	-	95.91	58.98

Conclusion

The defined objectives of this work are to reimplement the KVĚTA data-driven approach, train the BiLSTM-CRF sequence tagging model with various proposed input configurations, and decide – based on the obtained results – whether using the BiLSTM-CRF model for the metrical tagging of Czech syllabotonic verse is successful and has some benefits over using the KVĚTA data-driven approach.

Regarding the KVĚTA approach, it is reimplemented, but with minor changes to the original functionality, so the obtained predictions are probably a bit worse than they would be with the original program.

The BiLSTM-CRF model is successfully trained on many different input configurations, and with the best input configurations, it returns better results than the KVĚTA reimplementation with respect to all evaluated accuracies (syllable-level, line-level, and poem-level). Many observations about the BiLSTM-CRF input are made such that, not surprisingly, the syllable input yields better results than the token input (especially for the poem-level accuracy), but results for the token input, which does not need prior syllabification, are promising (particularly for the line-level accuracy). Another observation is that inputting a lemma significantly improves results for the token input, and the more additional features inputted (author of a poem, year of publication, POS tag, lemma), the better the predictions. The most interesting finding is that the best results are obtained by inputting sequences representing whole poems with line indices on the input that allow a model to distinguish different lines (especially the poem-level accuracy is significantly improved). The approach of inputting sequences representing whole poems may never have been tested before.

Overall, using the BiLSTM-CRF model for metrical tagging of Czech syllabotonic verse represents a great success and has many benefits over using the KVĚTA approach. It does not need to encode any complicated expert knowledge, as everything is learnt automatically by the machine learning model. The machine learning model can even encode its expert knowledge that humans do not understand. Furthermore, it is completely automatic, unlike the KVĚTA approach, which sometimes needs human assistance. On the other hand, with the KVĚTA approach, one has the confidence that for every poem, the returned metrical pattern is justified.

In the future, further experiments could be performed with the BiLSTM-CRF model, such as fine-tuning the model hyperparameters (optimizer with its settings, number of BiLSTM layers, number of recurrent units in one BiLSTM layer …), using character-based representations of the word embeddings, or training and using different word embeddings (e.g. GloVe, FastText, ELMo …). Syllabification without the need for phonetic transcription could be tried either by using hyphenation approaches or by training own machine learning model. Another interesting experiment could be multitask learning (e.g. predicting the metrical pattern and the POS tag together), predicting the name of the metre (dactyl, trochee …) instead of or alongside the metrical pattern in multitask setup, or transfer learning between poetic corpora in different languages. Furthermore, other machine learning architectures (e.g. transformers) could be tested.

Bibliography

1. PLECHÁČ, Petr; KOLÁR, Robert. The Corpus of Czech Verse. *Studia Metrica et Poetica*. 2015, vol. 2, no. 1, pp. 107–118. Available from DOI: 10.12697/smp.2015.2.1.05.
2. AGIRREZABAL, Manex; ALEGRIA, Iñaki; HULDEN, Mans. A Comparison of Feature-Based and Neural Scansion of Poetry. In: *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*. Varna, Bulgaria: INCOMA Ltd., 2017, pp. 18–23. Available from DOI: 10.26615/978-954-452-049-6_003.
3. IBRAHIM, Robert. *Teorie literatury: učebnice pro střední školy*. Plzeň: Fraus, 2014. ISBN 978-80-7238-926-1.
4. IBRAHIM, Robert; PLECHÁČ, Petr; ŘÍHA, Jakub. *Úvod do teorie verše*. Praha: Akropolis, 2013. ISBN 978-80-7470-051-4.
5. PLECHÁČ, Petr. Czech Verse Processing System KVĚTA – Phonetic and Metrical Components. *Glottotheory*. 2016, vol. 7, no. 2, pp. 159–174. Available from DOI: 10.1515/glot-2016-0013.
6. PLECHÁČ, Petr. *versotym/corpusCzechVerse 1.0*. Zenodo, 2021. Version 1.0. Available from DOI: 10.5281/zenodo.4569929.
7. PLECHÁČ, Petr; KOLÁR, Robert. Korpus českého verše. *Sborník semináře o digitálních zdrojích a službách ve společenskỳch a humanitních vědách. Praha: Ústav formální a aplikované lingvistiky MFF UK*. 2015, pp. 74–77.
8. HAJIČ, Jan. *Disambiguation of rich inflection: computational morphology of Czech*. Praha: Karolinum, 2004. ISBN 80-246-0282-2.
9. *SAMPA / X-SAMPA Transcription of Czech* [online]. 2015 [visited on 2022-02-25]. Available from: https://fonetika.ff.cuni.cz/o-fonetice/foneticka-transkripce/czech-sampa/.
10. *Corpus of Czech Verse, PHoEBE phonetic transcription* [online]. 2022 [visited on 2022-02-25]. Available from: https://versologie.cz/v2/web_content/phoebe.php.
11. HUANG, Zhiheng; XU, Wei; YU, Kai. Bidirectional LSTM-CRF Models for Sequence Tagging. 2015. Available from DOI: 10.48550/ARXIV.1508.01991.
12. LAFFERTY, John D.; MCCALLUM, Andrew; PEREIRA, Fernando C. N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: *Proceedings of the Eighteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 282–289. ICML '01. ISBN 1558607781.
13. REIMERS, Nils; GUREVYCH, Iryna. Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks. 2017. Available from DOI: 10.48550/ARXIV.1707.06799.
14. MIKOLOV, Tomás; CHEN, Kai; CORRADO, Greg; DEAN, Jeffrey. Efficient Estimation of Word Representations in Vector Space. In: BENGIO, Yoshua; LECUN, Yann (eds.). *1st*

International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings. 2013. Available also from: `http://arxiv.org/abs/1301.3781`.

15. PETERS, Matthew E.; NEUMANN, Mark; IYYER, Mohit; GARDNER, Matt; CLARK, Christopher; LEE, Kenton; ZETTLEMOYER, Luke. Deep Contextualized Word Representations. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 2227–2237. Available from DOI: `10.18653/v1/N18-1202`.
16. SOJKA, Petr; SOJKA, Ondřej. Towards Universal Hyphenation Patterns. In: ALEŠ HORÁK Pavel Rychlý, Adam Rambousek (ed.). *Proceedings of the Thirteenth Workshop on Recent Advances in Slavonic Natural Language Processing, RASLAN 2019*. Brno: Tribun EU, 2019, pp. 63–68. ISBN 978-80-263-1517-9. Available also from: `http://raslan2019.nlp-consulting.net/`.
17. BATCHELDER, Ned. *hyphenate.py* [online]. 2007 [visited on 2022-04-10]. Available from: `https://nedbatchelder.com/code/modules/hyphenate.py`.
18. HAIDER, Thomas. Metrical Tagging in the Wild: Building and Annotating Poetry Corpora with Rhythmic Features. In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, 2021, pp. 3715–3725. Available from DOI: `10.18653/v1/2021.eacl-main.325`.
19. IBRAHIM, Robert; PLECHÁČ, Petr. Toward Automatic Analysis of Czech Verse. *Formal Methods in Poetics. Lüdenscheid: RAM*. 2011, pp. 295–305.
20. UKPLAB. *emnlp2017-bilstm-cnn-crf: BiLSTM-CNN-CRF architecture for sequence tagging* [online]. GitHub, 2018 [visited on 2022-05-01]. Available from: `https://github.com/UKPLab/emnlp2017-bilstm-cnn-crf`.
21. SOJKA, Petr; SOJKA, Ondřej. The Unreasonable Effectiveness of Pattern Generation. *Zpravodaj CSTUG*. 2019, vol. 29. ISSN 1211-6661. Available from DOI: `http://dx.doi.org/10.5300/2019-1-4/73`.
22. LIANG, Franklin Mark. *Word Hy-Phen-a-Tion by Com-Put-Er (Hyphenation, Computer)*. Stanford, CA, USA: Stanford University, 1983. PhD thesis. AAI8329742.
23. *Pyphen* [online]. 2021 [visited on 2022-04-27]. Available from: `https://pyphen.org/`.

Contents of enclosed CD

- `README.md` .... description of the contents and run instructions
- `text` .... thesis text files
  - `thesis.pdf` .... thesis text in PDF format
  - `src` .... thesis text source code in LaTeX
- `requirements.txt` .... Python packages installation file
- `src` .... implementation