

# Comprehensibility and Automation: Plain Language in the Era of Digitalization

**István Üveges**

University of Szeged  
Doctoral School in Linguistics  
Egyetem utca 2  
Szeged 6722, Hungary  
uvegesistvan898@gmail.com

MONTANA Knowledge Management Ltd.  
Szállító u. 6  
Budapest 1211, Hungary  
uvegesi@montana.hu

**Abstract:** The current article briefly presents a pilot machine-learning experiment on the classification of official texts addressed to lay readers with the use of support vector machine as a baseline and fastText models.

For this purpose, a hand-crafted corpus was used, created by the experts of the National Tax and Customs Administration of Hungary under the office's Public Accessibility Programme. The corpus contained sentences that were paraphrased or completely rewritten by the experts to make them more readable for lay people, as well their original counter pairs. The aim was to automatically distinguish between these two classes by using supervised machine-learning algorithms.

If successful, such a machine-learning-based model could be used to draw the attention of experts involved in making the texts of official bodies more comprehensible to the average reader to the potentially problematic points of a text. Therefore, the process of rephrasing such texts could be sped up drastically.

Such a rephrasing (considering, above all, the needs of the average reader) can improve the overall comprehensibility of official (mostly legal) texts, and therefore supports access to justice, the transparency of governmental organizations and, most importantly, improves the rule of law in a given country.

**Keywords:** *access to justice, applied linguistics, comprehensibility, machine learning, plain language*

# 1. Introduction

Transparency and predictability of the law is a requirement of the rule of law. This kind of predictability is particularly pronounced in situations where lay audiences who do not speak a specialized language encounter official, often legal documents addressed to them by public authorities (Dobos, 2015; Vinnai, 2018; Tóth, 2019).

Plain language movement, which originates primarily from the US, and from the middle of the last century, aims to promote this transparency for non-expert recipients when it comes to legal/official language. This is particularly important in situations where the addressees of a text are lay people, who do not have the relevant domain knowledge in the given field. In such situations, if the text is not only based on technical knowledge, but is also over-complicated in its use of language and not considering the essential needs of an average reader, it even becomes difficult to determine exactly what information is not understood in a given text. In other words, (unnecessary) linguistic complexity and vagueness basically shadows the essential content.

Plain language supporters argue that by reducing such unnecessary linguistic complexity, the overall quality of a text can be significantly improved. To achieve this goal, they suggest and provide linguistic changes which, in their opinion, aids a text's general accessibility.

However, this kind of control is an extremely difficult task in practice and can only be carried out by experts in the given field (in other words, the solution is highly and inherently domain-specific). The aim of the present study is twofold. On the one hand, it seeks to experimentally answer the question of whether, given the right training data, one can interpret comprehensibility as a classification problem; the results obtained may provide some guidance in this respect. On the other hand, if such an interpretation proves feasible, it is possible to construct a machine-learning model that can be used to support the work of experts by labelling sentences that need special attention in texts that are about to be reformulated to make them more understandable for non-expert readers. However, such a simple classification does not shed any light on the exact cause of linguistic complexity of sentences but can make the experts' work much more effective.

In a later phase, sentences that were marked by the classifier as problematic could be subjected to a second check with the help of hand-crafted linguistic rules to get actual suggestions on 'how' to make them more comprehensible.

## 2. On the rule of law in Hungary

As mentioned above, the texts on which the machine-learning experiment in this paper is based were prepared with the assistance of linguistic experts from the National Tax and Customs Administration of Hungary. As is commonly known, a series of concerns about the state of the Hungarian Rule of Law have arisen in recent years, and have now led even to the launch of the so-called conditionality mechanism. The serious concerns and findings expressed by the Committee and other organizations (e.g., Transparency International) affects almost all areas of the legal system. In such a context, questions may legitimately arise about the reliability of the data used in the current research and originating in Hungarian public institutes.

This article draws heavily on the findings of the plain language movement (PLM), which is designed to promote transparency and legal certainty. The texts used are not, however, legal documents in the strict sense, but rather informational texts specifically aimed at the layperson, for example to help clarify the use of certain online interfaces of the tax administration, to explain how to submit a claim, or to answer common questions about case management. The article itself uses tools from computational linguistics and artificial intelligence research to solve a problem of a fundamentally linguistic nature.

In the light of the above, it is the author's firm view that these problems do not affect the investigation itself, neither the texts on which this study is based.

## 3. Literature

The comprehensibility of legal/official texts has been examined from many different perspectives over the past decades. Without claiming to be exhaustive, among the historically earliest methods, it is worth mentioning the use of readability formulas. Proponents of these formulas attempted to draw conclusions about a text's acceptability based on many quantitative properties of the sentences, paragraphs, words, etc. Such properties were, for instance, the ratio of the average number of syllables in a word, or the average number of words in a sentence (Dale & Chall, 1948; Dubay, 2004; Üveges, 2020). Readability formulas were in their heyday in the middle

of the last century, but their widespread use is greatly illustrated by the fact that they can still be found in (the English version of) MS Word as an optional indicator of text quality.

A significantly different approach is the way psycholinguistics deals with the issue of intelligibility from a cognitive perspective. The position of the psycholinguistics literature (Pléh & Lukács, 2014; Kas & Lukács, 2012; Pléh, 2013) offers a much less mechanical solution, which therefore may not necessarily be well amenable to automation. Based on their view, the factors that hinder comprehensibility are not located at the level of the sentence, but rather at the “global” level of the pragmatic context and the whole text. Therefore, psycholinguists mainly claim that only a few aspects can be detected within the boundaries of the sentence (such as the interruption of the main clause, or the atypical positioning of the main parts of the sentence, such as the subject, predicate and object).

The plain language movement, as stated above, was started in the US in the 1970s. Proponents of PLM approaches the making of official texts more accessible mainly from the aspect of improving the efficiency of communication and try to propose solutions at all levels of the text, such as typography, pragmatics, syntax, lexicon, etc. (Tiersma & Solan, 2012; Asprey, 2003). Among others, as a result of PLM, the Plain Language Act entered into force in the United States more than a decade ago. The Act regulates US government agencies’ information materials and communication with respect to their language use in accordance with plain language criterions (cf. Felsenfeld *et al.*, 1981; Cutts, 1999; Garner, 2001; Willerton, 2015).

In the Hungarian literature (linguistics and also theory of law), the lay perspective in a general sense has mainly come to the fore in the context of ‘law and language’ research. Its focus is typically on the idea that the basic pillar of equal access to law is (among other, mostly sociological factors) the requirement of equally accessible wording of legal texts (Vinnai, 2014; Szabó & Vinnai, 2018; Minya & Vinnai, 2018).

A closely related concept here is ‘access to justice’, which highlights the fact that, although the promulgation of legislation means that the body of law and the means of enforcing it are (in principle) equally available to anyone, but (due to sociological factors) this opportunity is not equally utilizable for everyone in practice. Law and language studies in Hungary often take as a starting point the fact that the requirement to ensure a fair trial and access to justice also implies the need that lay participants in the legal process

should not be put in a vulnerable, subordinate position. Such a situation can may occur simply because they do not understand the (technical) language in which professional participants are speaking to them. Therefore, the exact language that professionals use in situations when communicating with layperson are as important to access to justice as any other factors traditionally considered crucial for achieving and maintaining equal access to justice for everyone.

Even though this core idea is not widespread, similar thoughts from a slightly different perspective can be found in the literature. Solarte-Vasquez (2016; 2020a; 2020b), for instance, states that readability can be considered as a factor of usability and plain language as one of its criteria, and connects these concepts also to the notion of access to justice.

## 4. Methodology

Predictive binary classification, as a main method, was used here to solve the problem of the above-mentioned sentence classification. In this kind of machine-learning task, one must prepare a collection of data, in which individual instances are labeled (in our case, the labels are ‘original’ and ‘rephrased’ for sentences that were considered comprehensible by experts, and the ones that needed to be rephrased, respectively). The goal of the algorithms developed to solve classification problems is to predict labels on instances that they have never seen during the training phase (when the algorithm has to “learn” features associated with individual labels).

Such methods are widely used in various NLP tasks like sentiment analysis (e.g., Liu, 2012; Cabanlit & Espinosa, 2014) or part-of-speech tagging (Chiche & Yitagesu; 2022).

After that, in the case of NLP, the textual data should be transformed into a vector-based representation, since machine-learning algorithms work on numerical data. This can be solved in different ways, but the most common ones are using word frequencies in the text (like TF-IDF or bag of words—BoW), or word-embedding, created by neural networks on large corpora.

As mentioned, in order to successfully perform a supervised machine-learning experiment, the availability of relevant training data is inevitable. The Media Department of the Communications Department of the National Tax and Customs Administration of Hungary has been running a

Comprehensibility Programme for three years now. Under the programme, three (linguistic) experts monitor all the communication materials of the office and make suggestions and corrections regarding to plain language use. The reviewed documents are then published after a final peer review, again, carried out by legal/domain experts; these are going to be the final (publicly available) version, so they are not subject to a further re-check by linguistic experts.

Therefore, during this process, an original version and a rephrased variant exist of the same text, from which the latter is the result of applying (mostly) plain language principles. Based on these two versions, one can make training data suitable for a machine-learning algorithm designed to solve classification problems.

For this current research, the Office made available both versions from all their documents created in 2021 (in MS Word format). It is important to underline that the document versions used in the experiment (in accordance with the above) include the texts before the final peer review (carried out by legal/domain experts) and their original versions.

The files were available in groups of three, containing the original version, the redrafted version and a technical version in which corrections can be tracked.

#### 4.1 Data preparation and preliminary steps

The classification was based on the sentence level, as this is the level at which most of the literature suggests comprehensibility advice, and also the linguistic level that can be most reliably tested with automated tools.

In order to obtain a machine-readable format to be organized as pre-processed train and test sets, several processing steps had to be carried out. Initially, the MS Word files were converted into a .txt format, after which the texts were sentence segmented.

For the latter purpose, there are multiple potential solutions, which can be used in case of Hungarian language. Nonetheless, it is worth mentioning that sentence segmentation of legal texts in Hungarian cannot be considered as an already solved problem, so some experimentation for finding the best solution with the available tools is necessary. In terms of general-purpose language-processing tools, Hungarian is in a fairly good position, since there are several automatic parsing tools with sentence segmentation

modules. One of them is the magyarlanc linguistic processing toolkit (RGAI, n.d.; Zsibrita *et al.*, 2013), developed by the MTA-SZTE Research Group on Artificial Intelligence at the University of Szeged in the early 2010s. Another option, for instance, is the use of the HuSpaCy package (HuSpaCy, 2022; Orosz *et al.*, 2022), which is currently under development in the Artificial Intelligence National Laboratory (MILAB). These approaches are mainly based on machine-learning solutions, however the magyarlanc also has a built-in morphology. Another potential approach for handling sentence segmentation is to apply a heuristic solution, as the sentence-splitter package does (Text to sentence splitter, n.d.).

Preliminary investigations have shown that although machine-learning-based solutions are considered the state-of-the-art when it comes to natural language processing (NLP) tasks, they do not perform better than the mentioned heuristic approach on this specific corpus. In practice, it became clear that magyarlanc and HuSpaCy both made some typical mistakes while segmenting the original texts into sentences. One frequent source of error was the over segmentation of legal references, listings, and other special ways of structuring text that characterize the legal domain in general.

A big advantage of sentence-splitter is that it has a customizable list of exceptions that can be used to specify the abbreviations along which you want the tool not to segment sentences. In the present case, the list of abbreviations commonly applied in Hungarian legislation was used; it is publicly available online at the website of the Hungarian Supreme Court (*Kúria*, n.d.).

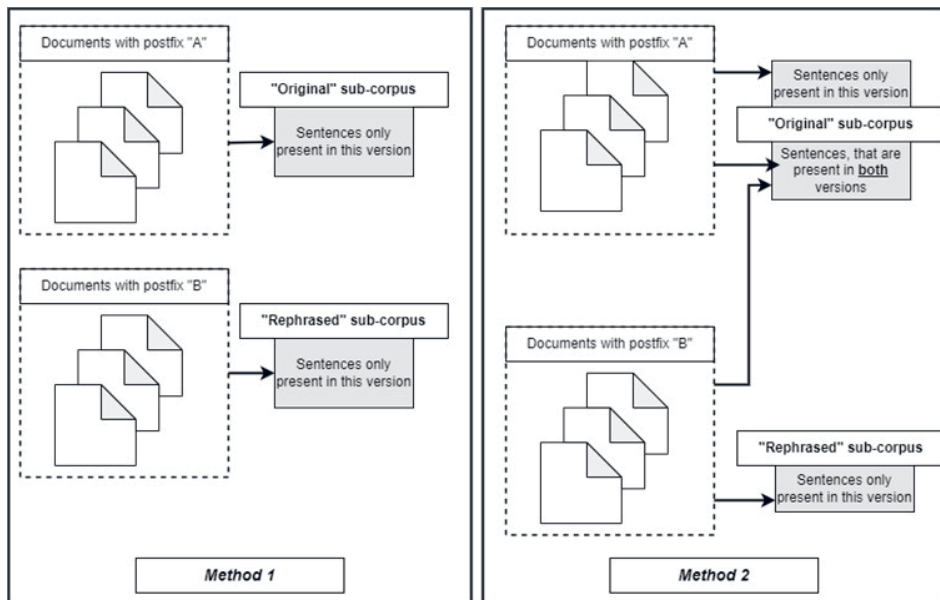
After the segmentation, the original corpus contained 40,057 sentences. A next step was to determine the difference between the two versions: the original and the rephrased texts. The naming convention of the original MS Word documents was a helpful source here, since it encoded

- if two documents are a version of the same text, and
- which is the original and which is the reviewed version.

For instance, in case of a document triplet; A1A.docx, A1B.docx and A1C.docx, the prefix (A1) encoded that they are versions of the same document, and the A, B and C postfixes encoded the status of the content. Here, A indicated that it is an original text, B showed that it is a rephrased version of the same text, while C indicated that it is a reviewed version, in which the proofreading can be followed.

A python script automatically found the document pairs (with A and B postfixes) and then determined the sentences which are only present in an original document (original sub-corpus) and the ones which are only present in the rephrased version (rephrased sub-corpus). This is illustrated as Method 1 in Figure 1.

**Figure 1.** Possibilities of sub-corpus selection



Another potential solution could have been the selection of all sentences present in both documents (original plus rephrased), then adding these to the ones that are present only in the original documents. This set would have been the 'comprehensible' sub-corpus, while the set of 'rephrased' sentences would have been only the ones that remained unchanged (Method 2 in Fig. 1).

Here, Method 1 was chosen because it selects a much narrower set from the texts to the original sub-corpus. This is an important factor, since, as was stated in the introduction, in our case, the sentences labelled as 'original' are in fact the ones that we want to label as problematic. Method 2 would therefore be better suited to construct a corpus to help us only separate original and rephrased sentences, but the subject of this article is more than this kind of pure separation.



## 4.2 Data selection and preprocessing

One last step was to remove (or at least reduce) the noise from the data. In this particular case, this meant removing incorrectly segmented sentences, and the ones which are indeed real sentences, but do not carry much information (e.g., falsely segmented listing tags, titles, footnotes, etc.). After the manual investigation of the data, the most effective and simple solution seemed to be to remove the sentences that are less than 10 tokens long.

**Figure 2.** Distribution of sentence length in terms of tokens in original and rephrased sub-corpora

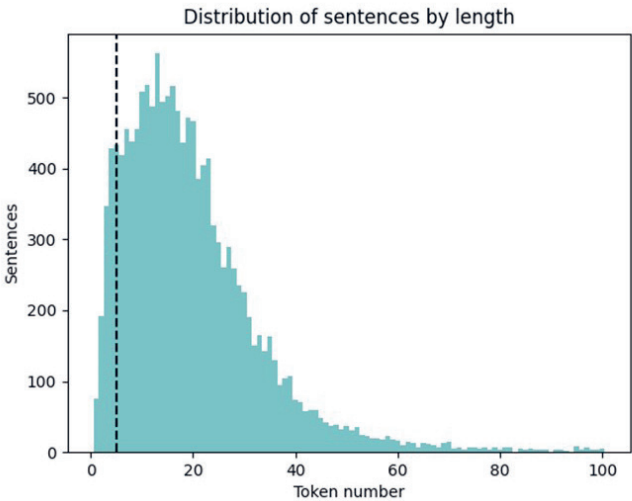


Figure 2 indicates the distribution of sentence lengths in the two sub-corpora (original and rephrased). It should be mentioned that the longest segmented sentence was 304 tokens long, but since such extremes were very rare, the Figure only includes data from the range of [1, 100] token length interval. The vertical dashed line indicates the 10 token boundary in sentence length.

Some basic properties of the remaining corpus can be seen in Table 1. The last row indicates the percentage of the sub-corpus related to the full dataset. From the original 40,057 sentences, the selected corpus contained 10,883 sentences<sup>1</sup>, which almost perfectly balanced between original and

<sup>1</sup> Compared to the original data, 14,123 sentences were changed during the expert review by the National Tax and Customs Administration. This means that 35.26% of the original text was slightly modified, the rest remained unchanged. About 77.06% of the modified sentences remained in the final corpus after the restriction of sentences to longer than 10 tokens.

rephrased sentences, therefore being theoretically optimal for a classification experiment.

**Table 1.** Basic statistics of the selected (final) corpus

|                         | Original | Rephrased |
|-------------------------|----------|-----------|
| Number of sentences     | 5,445    | 5,438     |
| Number of tokens        | 139,704  | 127,945   |
| Average sentence length | 25.66    | 23.53     |
| Percentage              | 50.03%   | 49.97%    |

As a final step, standard pre-processing took place before the model training could start, including lowercasing, removing numbers and punctuations, and then lemmatization<sup>2</sup>. This latter process was carried out with the help of the Hungarian language model of spaCy (HuSpaCy, n.d.).

## 5. Experiment I: support vector machine

After data preparation, a simple experiment was carried out with the use of scikit-learn support vector machine algorithm (scikit-learn, n.d.) with hyperparameter finetuning to determine the model that performs best in the available data.

Machine-learning algorithms generally have two sets of parameters; the learning (or estimation) of *model parameters* is part of the machine-learning process, while *hyperparameters* cannot be learned directly from the data, so their optimization can be done only manually. Despite this latter property, their optimal setting can have a major impact on the performance of the algorithm. In the current experiment, the following hyperparameters were used for optimization:

- C (regularization which controls the size of the margin of the hyperplane separating each class);
- gamma (which affects how far points from the potential separation boundary are still involved in the decision);
- kernel function (which controls the equation of the way the hyperplane is calculated of the equation of the hyperplane).

<sup>2</sup> Lemmatization is particularly important in the case of so-called morphologically rich languages like Hungarian, and it includes the normalization of inflected form into a single “stem”, but unlike stemming, the process requires contextual information about the given word, not just the inflected form itself.

In addition, TF-IDF vectorization (Sammut & Webb, 2011) was used to transform the data (the sentences) to a final, machine-readable format.

The mentioned hyperparameters were set according to all possible combinations (4<sup>3</sup>) of the following values:

- C: [0.1, 1, 10, 100]
- gamma: [1, 0.1, 0.01, 0.001]
- kernel: ['rbf', 'poly', 'sigmoid', 'linear'].

Since the training set used is relatively small, the usual metrics (precision, recall, and F1) were calculated after 10-fold cross-validation.

**Table 2.** Results of the 10 best models after hyperparameter finetuning (P: Precision, R: Recall, F1: F-Score, AVG: average during 10-fold cross-validation, STD: Standard deviation during 10-fold cross-validation)

| Parameters                             | Class     | P<br>(AVG) | P<br>(STD) | R<br>(AVG) | R<br>(STD) | F1<br>(AVG) | F1<br>(STD) | F1 (AVG -<br>2 class) |
|--|-----------|------------|------------|------------|------------|-------------|-------------|-----------------------|
| C: 10, gamma: 1,<br>kernel: linear     | original  | 0.66       | 0.02       | 0.69       | 0.05       | 0.68        | 0.04        | 0.68                  |
|  | rephrased | 0.68       | 0.03       | 0.65       | 0.03       | 0.66        | 0.02        |                       |
| C: 10, gamma: 0.1,<br>kernel: linear   | original  | 0.66       | 0.02       | 0.69       | 0.05       | 0.68        | 0.04        | 0.68                  |
|  | rephrased | 0.68       | 0.03       | 0.65       | 0.03       | 0.66        | 0.02        |                       |
| C: 10, gamma: 0.01,<br>kernel: linear  | original  | 0.66       | 0.02       | 0.69       | 0.05       | 0.68        | 0.04        | 0.68                  |
|  | rephrased | 0.68       | 0.03       | 0.65       | 0.03       | 0.66        | 0.02        |                       |
| C: 10, gamma:<br>0.001, kernel: linear | original  | 0.66       | 0.02       | 0.69       | 0.05       | 0.68        | 0.04        | 0.675                 |
|  | rephrased | 0.68       | 0.03       | 0.65       | 0.03       | 0.66        | 0.02        |                       |
| C: 100, gamma: 0.1,<br>kernel: rbf     | original  | 0.66       | 0.03       | 0.68       | 0.05       | 0.67        | 0.03        | 0.67                  |
|  | rephrased | 0.68       | 0.03       | 0.65       | 0.02       | 0.67        | 0.02        |                       |
| C: 100, gamma: 0.1,<br>kernel: sigmoid | original  | 0.66       | 0.02       | 0.69       | 0.05       | 0.67        | 0.03        | 0.67                  |
|  | rephrased | 0.68       | 0.03       | 0.65       | 0.03       | 0.67        | 0.02        |                       |
| C: 100, gamma:<br>0.01, kernel: rbf    | original  | 0.68       | 0.03       | 0.66       | 0.07       | 0.67        | 0.05        | 0.66                  |
|  | rephrased | 0.67       | 0.04       | 0.68       | 0.03       | 0.67        | 0.03        |                       |
| C: 1, gamma: 1,<br>kernel: sigmoid     | original  | 0.69       | 0.04       | 0.61       | 0.06       | 0.65        | 0.05        | 0.65                  |
|  | rephrased | 0.65       | 0.03       | 0.73       | 0.04       | 0.69        | 0.03        |                       |
| C: 1, gamma: 1,<br>kernel: linear      | original  | 0.68       | 0.04       | 0.62       | 0.06       | 0.65        | 0.05        | 0.65                  |
|  | rephrased | 0.65       | 0.04       | 0.71       | 0.05       | 0.68        | 0.03        |                       |
| C: 1, gamma: 0.1,<br>kernel: linear    | original  | 0.68       | 0.04       | 0.62       | 0.06       | 0.65        | 0.05        | 0.65                  |
|  | rephrased | 0.65       | 0.04       | 0.71       | 0.05       | 0.68        | 0.03        |                       |

Table 2 represents the results of the 10 best models for separating original sentences from rephrased ones. The results show that, in general, the best kernel function seems to be the ‘linear’ kernel (it characterizes 6 out of 10 best models), while regularization (C) seems to be the best if set to 10 (top 4 models), while ‘gamma’ shows no clear tendency (it varies widely across the selected models). The selected models seem balanced in finding the original and the already rephrased sentences also.

Another important question is which models can most accurately identify the original sentences (since the main goal is to identify problematic points in the text to make it easier to identify these points for the experts). Table 3 reflects this aspect by indicating the top 5 models in identifying original sentences (ranked by Precision of the predictions related to original sentences in the test sets).

**Table 3.** Best fitting models for predicting sentences that should be rephrased.

| Parameters                          | P<br>(AVG) | P<br>(STD) | R (AVG) | R<br>(STD) | F1<br>(AVG) | F1 (STD) |
|-------------------------------------|------------|------------|---------|------------|-------------|----------|
| C: 0.1, gamma: 1, kernel: rbf       | 0.79       | 0.28       | 0.03    | 0.03       | 0.05        | 0.06     |
| C: 0.1, gamma: 1, kernel: sigmoid   | 0.72       | 0.05       | 0.45    | 0.08       | 0.55        | 0.07     |
| C: 0.1, gamma: 1, kernel: linear    | 0.71       | 0.04       | 0.47    | 0.08       | 0.57        | 0.07     |
| C: 0.1, gamma: 0.1, kernel: linear  | 0.71       | 0.04       | 0.47    | 0.08       | 0.57        | 0.07     |
| C: 0.1, gamma: 0.01, kernel: linear | 0.71       | 0.04       | 0.47    | 0.08       | 0.57        | 0.07     |

The two sets of models are almost completely overlapping instead of the best model, which get an average 0.79 Precision score. However, this model has the biggest standard deviation (0.28) among all, which means that this precision value is highly unstable.

The other models seem much more reliable, therefore a better choice for practical use. To sum up, optimal hyperparameters seem to be 0.1 in case of regularization, and 1 regarding gamma. Kernel function varies in the case of the best models, but in the majority of them, it is the linear kernel.

## 6. Experiment II: fastText

FastText is a relatively new (open source) library for learning word embeddings, and it can also be used for supervised text classification. The algorithm was originally created by the Facebook AI Research (FAIR) lab (Joulin *et al.*, 2016). Its big practical advantage compared to other neural network-based solutions is its ease of use (it does not require deep mathematical knowledge beyond the initialization of the parameter sets of the models). In return, however, it does not provide as much customization possibilities as, for instance, scikit-learn LSTM (where settings can be made even for the properties of the individual layers in the constructed neural network).

While in the case of SVM, the vectorized representation was based on TF-IDF, fastText working with word embeddings, which we can get from an initial neural network by feeding the one-hot encoded representation of the given word with the network, and by extracting the hidden layers after the process (in case of skip-grams).

There are pretrained vectors available for 157 different languages (fastText, n.d., b), including languages that are traditionally considered less resourceful (like Hungarian or Estonian) compared to “big languages” like English in terms of NLP tools.

Another benefit of applying fastText is the subword-level representation it is using. While TF-IDF cannot handle OOV (out of vocabulary) words, that is words that were not present while training the vectorizer, fastText models can represent such new entries by assigning them representations from its underlying character n-grams.

### 6.1 Hyperparameter optimization

For the current research, both pretrained vectors for Hungarian<sup>3</sup> from the official website of fastText and vectors trained on the given corpora were used. The former has an advantage of having been trained in a much bigger dataset, while the latter can represent domain-specific, or even corpus-specific embeddings better.

<sup>3</sup> At the official webpage, the .bin format is available from the word vectors; however, when using pretrained vectors, the model needs a .vec format. The conversion can be carried out manually; a potential working solution can be found at the fastText’s Github repository’s forum (fastText, 2020).

The tested hyperparameters for the models are listed below:

- **Learning rate ('lr'):** A model's learning rate is a way of controlling how "fast" the model updates during the training phase. This parameter controls the size of the update that is applied to the parameters of the models. FastText's official page (fastText, n.d., a) recommends this to be set between 0.1 and 1.
- **Word n-grams ('wordNgrams'):** By setting this value to 1, we can define the size of word collocations (co-occurring tokens) to be considered during the training phase. In the simplest case, we can use just unigrams (individual tokens, but in complex problems like the comprehensibility of official texts, a higher value can also be beneficial.
- **Maximum epoch number of training ('epoch'):** By setting the maximum number of epochs, one can control how many times the model sees each training example during training. The default value is 5 but setting a higher number may also be appropriate.
- **Using pretrained embeddings, or ones that are trained on the actual dataset ('pretrainedVectors'):** There is both an option of using pretrained vectors (generated with a significantly greater amount of data) or ones that are trained only in the given corpus.
- **The minimum occurrence that a token must have in order to be considered during training ('minCount'):** fastText provides a parameter which is used to control the minimum number of occurrences in the corpus, under which a token is not considered during the classification.
- **Dimension of the used word-embeddings ('dim'):** By reducing the dimension of the word-embeddings, one can perform different experiments with word-vectors that have lower dimensions than fastText's default 300.

By setting these parameters into different values, we can earn an insight about how these settings influence the results of the models.

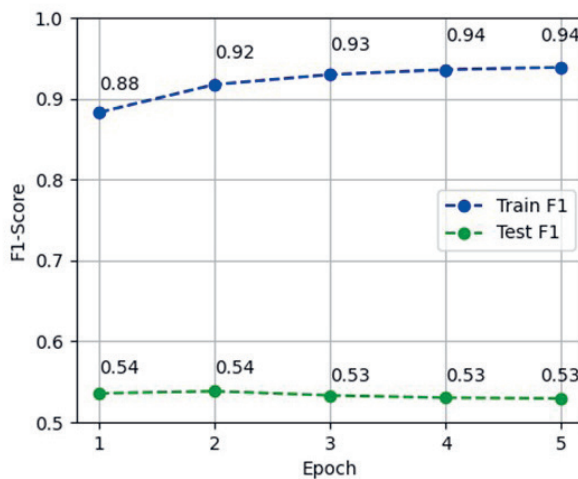
## 6.2 General results

As in the case of SVM, here the corpus was separated in an 80-20 percent ratio to train and test set. The preliminary hypotheses were that pre-trained word vectors should have performed better than the ones trained on just the actual corpus, and that n-grams that consider more tokens should have performed better than unigrams. On the other hand, there was no similar prior assumption for the values of the other parameters listed above.

Since all internal weights in the model are randomly initialized at every run, and fastText does not provide a way to control this behavior (like scikit-learn does with the random state option) it was useful to test every set of parameters with multiple runs per epoch. A value of 5 runs per epoch was selected, after which the F1-Scores obtained were averaged in order to obtain a realistic picture of the results.

After some preliminary experimentation with the pre-trained vectors (with only 5 epochs of training), it turned out that the models trained on these vectors were overfitted already at the first epoch. This kind of behavior was not affected significantly by changing any of the parameters mentioned above. Figure 3 illustrates a typical result after testing the model both on the train and test set with the use of pretrained word-vectors.

**Figure 3.** Measured F1-Scores with {"lr": 0.3, "wordNgrams": 1, "minCount": 1, "epoch": 5} parameter set



After it became clear that the pre-trained word vectors could not generalize over the data, corpus-trained vectors were used in the next phase.

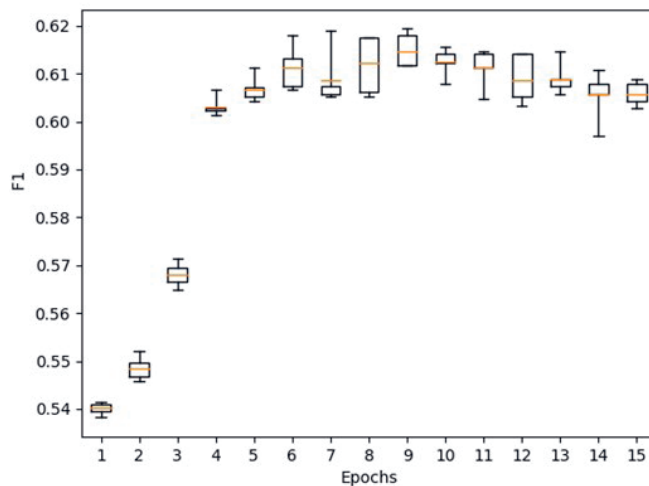
The following main conclusions were then drawn from the testing of different combinations of possible parameters:

- To avoid overfitting again, the learning-rate should be set to the recommended minimum value (0.1);
- Using n-grams instead of unigrams also worsened the test set results in all cases,

- The value that allowed tokens above a certain frequency of occurrence to be considered (minCount) was worth setting to 10 (although setting it to higher values, e.g., 20 or 30, did not significantly increase the results produced on the test set);
- Changing the number of dimensions of the used word embeddings did not seem to have a significant effect on the results, but seemed to trigger overfitting.

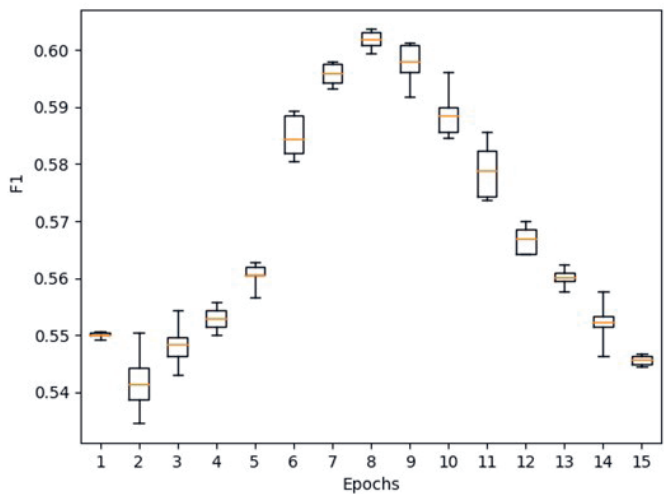
The best results were obtained by setting learning rate to 0.1, n-gram-size to 1, minCount to 20 and dimension of embeddings to 50 (later referred as “best parameter set”). The F1-Scores given are illustrated in Figure 4.

**Figure 4.** F1-Scores on train set with the best parameter-set (min, max, q1 and q3 quartiles, and average are represented with regard the average of 5 different runs per every epoch)



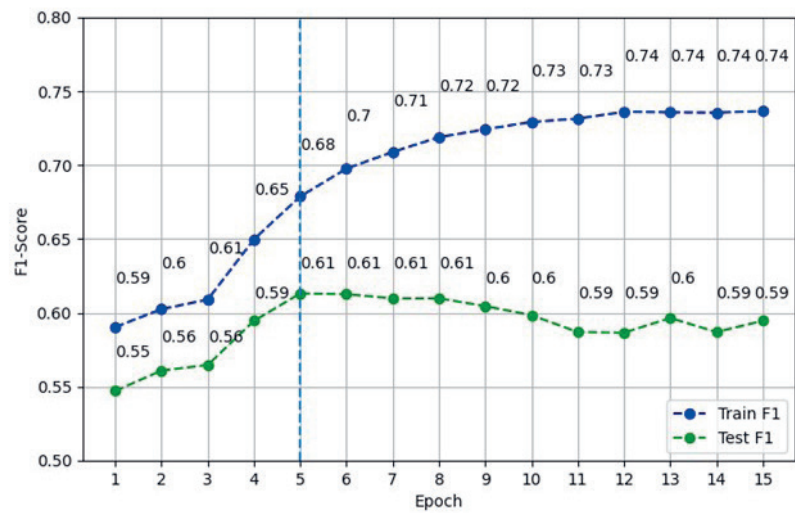


**Figure 5.** Changing the embedding dimension from 50 to 300, however, did not affect the best model (8 epochs) significantly, but highly triggered overfitting during the subsequent epochs



The effect of changing the embeddings' dimension is illustrated in Figure 5 where, again, the learning rate was set to 0.1, n-gram-size to 1, minCount to 20 but dimension of embeddings to 300 instead of 50.

**Figure 6.** F1-Scores measured in the train and test set with best parameter set



By comparing train and test F1-Scores (the average of 5 different runs) it can be seen that the optimal epoch number is 5 in this case (illustrated in Fig. 6). In other words, this is the number of epochs where the best score can be achieved in the train set, but the difference between train and test F1-s remains relatively small. After this point, F1 start to decrease slowly on the test set while it keeps rising on the train data, which means that the model was overfitted.

### 6.3 Lemmatization

Lemmatization is a commonly used technique in NLP when dealing with morphologically rich languages. Its major difference from stemming (commonly used in NLP solutions in the case of English) can be described by emphasizing that stemming mainly just strip suffixes from the end of the word (therefore not necessarily producing a meaningful word form) while lemmatization always results in a word's dictionary form (Jurafsky & Martin, 2021, p. 20).

Both approaches are used during the construction of the vector space model to bring the different forms of a single word into a common canonical form for representing them uniformly. This means that the information associated with the (inflected) word form is lost, but the dimension of the vector space can be significantly reduced. In English, this reduction can be as high as 40–70%, while in Hungarian it is reported to be as high as 90% (Tikk, 2007, p. 41). Whether the separate representation of inflected words is unnecessary noise or valuable information, is highly domain and application dependent.

In order to achieve a higher F1-Scores on the test set, a lemmatized form of the data was also tested with fastText models (with same hyperparameter sets as before). The lemmatization process was carried out again with the Hungarian language model of spaCy (see 4.2). The results showed that despite the drastic reduction in the number of isolated word forms (68.74% less), the precision and recall on the test set did not change significantly. Table 4 shows the results of two typical pairs of models, one of which received lemmatized text as input in all cases, while the other got texts prepared with the preprocessing steps mentioned earlier. Other parameters were identical in both cases. Here:

- Pair A refers to {"lr": 0.1, "wordNgrams": 1, "minCount": 20, "epoch": 10, "dim": **50**} setting, while
- Pair B to {"lr": 0.1, "wordNgrams": 1, "minCount": 20, "epoch": 10, "dim": **300**} parameters.

**Table 4.** Measured F1-Score comparison of 2 parameter sets in terms of the effect of lemmatization—5 runs / epoch’s average

| Epoch | Pair A            |        |                        |        | Pair B            |        |                        |        |
|-------|-------------------|--------|------------------------|--------|-------------------|--------|------------------------|--------|
|       | Simple preprocess |        | Simple + lemmatization |        | Simple preprocess |        | Simple + lemmatization |        |
|       | Train             | Test   | Train                  | Test   | Train             | Test   | Train                  | Test   |
| 1     | 0.5750            | 0.5402 | 0.5748                 | 0.5409 | 0.5890            | 0.5497 | 0.5899                 | 0.5511 |
| 2     | 0.5810            | 0.5486 | 0.5776                 | 0.5488 | 0.5883            | 0.5497 | 0.6033                 | 0.5686 |
| 3     | 0.6135            | 0.5679 | 0.6143                 | 0.5680 | 0.5955            | 0.5539 | 0.6052                 | 0.5620 |
| 4     | 0.6694            | 0.6031 | 0.6699                 | 0.6047 | 0.6416            | 0.5950 | 0.6060                 | 0.5575 |
| 5     | 0.7026            | 0.6067 | 0.7030                 | 0.6068 | 0.6890            | 0.6075 | 0.6392                 | 0.5872 |
| 6     | 0.7168            | 0.6113 | 0.7165                 | 0.6123 | 0.7132            | 0.6103 | 0.6685                 | 0.6024 |
| 7     | 0.7270            | 0.6085 | 0.7277                 | 0.6150 | 0.7223            | 0.6187 | 0.6961                 | 0.6059 |
| 8     | 0.7334            | 0.6121 | 0.7333                 | 0.6121 | 0.7315            | 0.6159 | 0.7119                 | 0.6056 |
| 9     | 0.7375            | 0.6146 | 0.7378                 | 0.6123 | 0.7362            | 0.6101 | 0.7304                 | 0.5981 |
| 10    | 0.7416            | 0.6125 | 0.7419                 | 0.6079 | 0.7400            | 0.6096 | 0.7439                 | 0.5930 |

By observing the results, it can be concluded that the difference between the two models’ F1-Scores is usually less than 0.01, which can be considered as negligible.

## 7. Conclusions

By attempting to train a machine-learning-based model for classifying the intelligibility of official texts, we are in fact also attempting to answer the question of the extent to which human intuitions about the intelligibility of these texts can be reproduced by machine-learning-based models.

It is important to stress, however, that, given the nature of the corpus available, the models trained for the current study are not in fact an algorithmic implementation of a general definition of intelligibility. Since there was no adequate training data available, either in terms of quantity or in terms of the diversity of the source and scope of the input texts, the current solution remains highly domain-specific. Therefore, the models produced here can be seen as an approximation of the intuitive notions of the concept of comprehensibility by NAV experts.

Taking all this into account, in the current pilot study, machine-learning experiments were carried out using support vector machine and fastText models. In both cases, the hyperparameters of each model were optimized to improve the achievable prediction performance.

Although traditional machine-learning models are now outperformed in most areas by models using word embeddings, this trend has not been reflected regarding the current corpus. Since fastText does not support the evaluation of the models per class, models were evaluated based on F1-Scores for two classes (original and rephrased). Here the best a model could achieve was  $\sim 0.62$ , which can be considered moderately effective. On the other hand, SVM models could achieve a stable  $\sim 0.72$  regarding identifying original sentences in the test set.

A comprehensive error-analysis is needed to define the potential crucial points on why the theoretically more modern solution performed worse on the actual data. Moreover, fastText is not the most customizable nor the most cutting-edge solution in the field of NLP.

Preliminary results suggest that, with the right choice of the classification model, automatic detection of problematic sentences (in respect of comprehensibility) can be identified, and this would be a big step towards supporting accessible administrative communication by machine-learning-based solutions.

## Acknowledgement

This research was supported by the ÚNKP-21-3 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund.

**István Üveges** is a PhD student in linguistics at the University of Szeged, and a participant in the Artificial Intelligence National Laboratory (MILAB) sentiment analysis sub-project at the Center for Social Sciences, Budapest, Hungary. His areas of interest include sentiment analysis, natural language processing, plain language movement and artificial intelligence (machine learning). He is working as a computational linguist researcher at MONTANA Knowledge Management Ltd., where he deals with legaltech issues, such as classification of legal documents with both supervised and unsupervised machine learning and various domain-specific IT solutions.

## References

- Asprey, M. M.** (2003), *Plain Language for Lawyers*, Annandale, N.S.W.: Federation Press.
- Cabanlit, M. A. & Espinosa, K. J.** (2014), 'Optimizing N-gram based text feature selection in sentiment analysis for commercial products in Twitter through polarity lexicons,' in *Information, Intelligence, Systems and Applications*, IISA 2014, The 5th International Conference, pp. 94–97.  
<https://doi.org/10.1109/IISA.2014.6878767>
- Chiche, A. & Yitagesu, B.** (2022), 'Part of speech tagging: A systematic review of deep learning and machine learning approaches,' *Journal of Big Data*, vol. 9, art. 10, pp. 1–25. <https://doi.org/10.1186/s40537-022-00561-y>
- Cutts, M.** (1999), *The Plain English Guide*, Oxford: Oxford University Press.
- Dale, E. & Chall, J. S.** (1948), 'A formula for predicting readability,' in *Educational Research Bulletin*, vol. 27, no. 1, pp. 11–20.
- Dobos, Cs.** (2015), 'Nyelven belüli fordítás és tisztességes jogi eljárás' [Intralingual translation and fair trial], in M. Szabó (ed.) *A jog nyelvi dimenziója* [The linguistic dimension of law], Miskolc, Magyarország: Bíbor Kiadó, pp. 215–226.
- Dubay, W. H.** (2004), *The Principles of Readability*, Costa Mesta: Impact Information.
- fastText (n.d., a), 'Text classification.' Retrieved from <https://fasttext.cc/docs/en/supervised-tutorial.html> [accessed Nov 2022]
- fastText (n.d., b), 'Word vectors for 157 languages.' Retrieved from <https://fasttext.cc/docs/en/crawl-vectors.html> [accessed Nov 2022]
- fastText (2020), 'How to turn .bin to .vec #1082,' *GitHub repository forum*. Retrieved from <https://github.com/facebookresearch/fastText/issues/1082> [accessed Nov 2022]
- Felsenfeld, C.; Cohen, D. S. & Fingerhut, M.** (1981), 'The Plain English movement in the United States: Comments,' *Canadian Business Law Journal*, vol. 6, pp. 408–452.
- Garner, B. A.** (2001), *Legal Writing in Plain English: A Text with Exercises*, Chicago: University of Chicago Press. <https://doi.org/10.7208/chicago/9780226284200.001.0001>
- HuSpaCy (n.d.), Core Hungarian model for HuSpaCy, *Hugging Face*. Retrieved from [https://huggingface.co/huspace/hu\\_core\\_news\\_lg](https://huggingface.co/huspace/hu_core_news_lg) [accessed Nov 2022]
- HuSpaCy (2022), HuSpaCy: An industrial-strength Hungarian natural language processing toolkit, *Github Repository*. Retrieved from <https://github.com/huspace/huspace> [accessed Nov 2022]
- Joulin, A.; Grave, E.; Bojanowski, P. & Mikolov, T.** (2016), 'Bag of tricks for efficient text classification,' *ArXiv preprint*. <https://doi.org/10.48550/arXiv.1607.01759>

- Jurafsky, D. & Martin, J. H.** (n.d.), *Speech and Language Processing*, 3rd ed. draft. Retrieved from <https://web.stanford.edu/~jurafsky/slp3/> [accessed 10 Jul 2022]
- Kas, B. & Lukács, A.** (2012), 'Processing relative clauses by Hungarian typically developing children,' *Language and Cognitive Processes*, vol. 27, no. 4, pp. 500–538. <https://doi.org/10.1080/01690965.2011.552917>
- Kúria** (n.d.), 'A magyar jogszabályok rövidítései' [Abbreviations used in Hungarian legislation]. Retrieved from [https://kuria-birosag.hu/sites/default/files/joggyak/3\\_melleklet.pdf](https://kuria-birosag.hu/sites/default/files/joggyak/3_melleklet.pdf) [accessed Nov 2022]
- Liu, B.** (2012), *Sentiment Analysis and Opinion Mining*, Synthesis Lectures on Human Language Technologies, no. 5, Cham: Springer. <https://doi.org/10.1007/978-3-031-02145-9>
- Minya, K. & Vinnai, E.** (2018), 'Hogyan írjunk érthetően? kilendülés a jogi szaknyelv komfortzónájából' [How to write clearly? Getting out of the comfort zone of legal jargon], *Magyar Jogi Nyelv* [Hungarian legal language], no. 1, pp. 13–18.
- Orosz, Gy.; Szántó, Zs.; Berkecz, P.; Szabó, G. & Farkas, R.** (2022), 'HuSpaCy: an industrial-strength Hungarian natural language processing toolkit,' in G. Berend, G. Gosztolya & V. Vincze (eds.) *XVIII. Magyar Számítógépes Nyelvészeti Konferencia* [Hungarian Computational Linguistics Conference], Szeged, Magyarország: Szegedi Tudományegyetem, Informatikai Intézet, pp. 59–73.
- Pléh, Cs.** (2013), *A lélek és a nyelv* [The soul and language], Budapest, Magyarország: Akadémiai Kiadó. <https://doi.org/10.1556/9789630596817>
- Pléh, Cs. & Lukács, Á.** (2014), *Pszicholingvisztika* [Psycholinguistics], Budapest, Magyarország: Akadémiai Kiadó. <https://doi.org/10.1556/9789630594998>
- RGAI** (n.d.), 'magyarlanc: a toolkit for linguistic processing of Hungarian,' MTA-SZTE Research Group on Artificial Intelligence. Retrieved from <https://rgai.inf.u-szeged.hu/magyarlanc> [accessed Nov 2022]
- Sammut, C. & Webb, G. I.** (2011), 'TF-IDF,' in C. Sammut & G. I. Webb (eds.) *Encyclopedia of Machine Learning*, Boston, MA: Springer. <https://doi.org/10.1007/978-0-387-30164-8>
- scikit-learn** (n.d.). Retrieved from <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html> [accessed Nov 2022]
- Solarte-Vasquez, M. C. & Hietanen-Kunwald, P.** (2020a), 'Transaction design standards for the operationalisation of fairness and empowerment in proactive contracting,' *International and Comparative Law Review / Mezinárodní a Srovnávací Právní Revue*, vol. 20, no. 1, pp. 180–200. <https://doi.org/10.2478/iclr-2020-0008>
- Solarte-Vasquez, M. C. & Hietanen-Kunwald, P.** (2020b), 'Responsibility and responsiveness in the design of digital and automated dispute resolution processes,' in *23rd International Legal Informatics Symposium (IRIS 2020)*,

- University of Salzburg, Salzburg, Austria, February 27–29, 2020, pp. 451–459.  
<https://doi.org/10.38023/2038d4dc-d497-49eb-9179-0d2c77f64132>
- Solarte-Vasquez, M. C.; Järv, N. & Nyman-Metcalf, K.** (2016), ‘Usability factors in transactional design and smart contracting,’ in T. Kerikmäe & A. Rull (eds.) *The Future of Law and eTechnologies*, Cham: Springer, pp. 149–176.  
[https://doi.org/10.1007/978-3-319-26896-5\\_8](https://doi.org/10.1007/978-3-319-26896-5_8)
- Text to sentence splitter (n.d.), *GitHub Repository*. Retrieved from <https://github.com/mediacloud/sentence-splitter> [accessed Nov 2022]
- Tiersma, P. M. & Solan, L. M.**, eds. (2012), *The Oxford Handbook of Language and Law*, New York: Oxford University Press.  
<https://doi.org/10.1093/oxfordhb/9780199572120.001.0001>
- Tikk, D.**, ed. (2007), *Szövegbányászat* [Text mining], Budapest, Magyarország: Typotex Kiadó.
- Tóth, J.** (2019), ‘Tudnak-e a jogászok érthetően fogalmazni, avagy nem is kell azt tudni?’ [Can lawyers write clearly, or do they not need to?] *Magyar Jogi Nyelv* [Hungarian legal language], no. 1, pp. 31–37.
- Üveges, I.** (2020), ‘Automatizálható a közérthető megfogalmazás? Jog, számítógépes nyelvészet és pszicholingvisztika találkozása’ [Can Plain Language be automated? Intersection between law, computational linguistics and psycholinguistics], *Magyar Jogi Nyelv* [Hungarian legal language], no. 1, pp. 1–8.
- Vinnai, E.** (2014), ‘A magyar jogi nyelv kutatása’ [Research on the Hungarian legal language], *Glossa Iuridica*, no. 1, pp. 29–48.
- Vinnai, E.** (2018), ‘Megértette a figyelmeztetést? A figyelmeztetések és tájékoztatások közlése a büntetőeljárásokban’ [Did you understand the warning? Communication of warnings and information in criminal proceedings], in M. Szabó & E. Vinnai (eds.) *A törvény szavai: Az OTKA-112172 kutatási zárókonferencia anyaga Miskolc* [Words of the law: Proceedings of the OTKA-112172 final research conference], *ME – MAB, 2018. május 25*, Miskolc, Magyarország: Bíbor Kiadó, pp. 281–295.
- Willerton, R.** (2015), *Plain Language and Ethical Action: A Dialogic Approach to Technical Content in the 21st Century*, New York: Routledge. <https://doi.org/10.4324/9781315796956>
- Zsibrita, J.; Vincze, V. & Farkas, R.** (2013), ‘magyarlanc: A toolkit for morphological and dependency parsing of Hungarian,’ in R. Mitkov, G. Angelova & K. Bontcheva (eds.) *Proceedings of RANLP 2013*, Hissar, Bulgaria: INCOMA Ltd., pp. 763–771.