Faculty of Computer Science

Master of Data Science Programme

Moscow, 2022

Developing framework (FuzzyText) for extracting structured data from unstructured fuzzy texts, using the concept of similar contexts

Student: Alen Rafagudinov

Supervisor: Doctor of Sciences, HSE Professor, Vasilii Gromov

Consultant: Ex-CTO Avito.ru, Roman Pavlushko

Reasons to extract structured data from texts

The text data is everywhere, it is probably the most significant data available for data scientists.

The problem is that many of the data science methods are supervised. These methods require labeled data, but it is not always available for texts.

Moreover, structured data can be required for many other tasks as well. For example, for filtering or sorting texts by some value.

So extracting structured information from unstructured raw texts is a widespread and essential task.

Issues of extracting data from internet texts

The biggest challenge of performing structured data extracting from internet texts is that such texts are vastly different from the standard language.

People often do not care about quality or format. There are grammar mistakes, abbreviations, slang.

So real-world texts can be messy, and dealing with such texts can be very time-consuming and painfully frustrating.

Popular approaches to deal with the issues

Regular expressions

A popular method to extract data from fuzzy texts is building a sophisticated set of rules using regular expressions.

Edit distance

Another popular way is using edit distance (e.g. Levenshtein) to find similar terms to the reference ones in a given text.

Issues of the regular expressions approach

This method requires to foresee as much as possible variants of spellings and contexts.

It often involves a considerable amount of manual work, which leads to very complex regular expressions that are difficult to maintain and improve.

Also, since we create these rules manually, we may skip many cases. Attempts to cover all of them using regular expressions may lead to an unmaintainable system.

Сдам **1-комнатную** квартиру

Сдается срочно отличная **1к** новая мебл. квартирка, в которой есть **2к** холодильник

Rental price: $**3000**

Nice townhouse, 1000 sq ft **3000** monthly

Сдам дом, 1000 кв.м, **50000** + к.у.

Issues of the edit distance based approach

The edit distance method requires significantly less manual work to resolve misspellings. However, it does not consider the context.

In the first example, both “1к” and “2к” can be valid values, so there is a collision without keeping the context in mind.

When it is required to extract numerical values depending solely on the context, this approach does not work at all.

Сдается **2-комнатная** квартирка
Сдается **2к** квартирка
Сдается **1к** квартирка, в которой есть **2к** холодильник

Rental price: $**3000**
Nice townhouse, 1000 sq ft **3000** + utilities

Setting hypothesis of the similar contexts

According to the distributional hypothesis, similar words tend to be in the same context. So intuitively, we can assume the reverse hypothesis, that **similar contexts tend to have the same words**.

If this hypothesis is correct, we can set in advance a reference context where the desired value may be located, and then find similar contexts (positions) in the analyzed text. It will give us the most probable position, where the sought-for value seats

Similar words tend to be in the same context

Similar contexts tend to have the same words

The concept of the similar contexts

Let's run some examples. Let's state we need to extract the number of rooms from a fuzzy Russian text. First, let's set up a reference context where the number of rooms is likely to be:

*Сдается * квартира*

Using the RoBERTa model trained on Russian rental ads, we get the 100 most probable tokens that can be in the place of the asterisk.

	Token	Probability
1	однокомнатная	0.62
2	двухкомнатная	0.12
3	новая	0.08
4	трехкомнатная	0.06
5	уютная	0.03
...	...	...
8	однакомнатная	0.01
...	...	...
100	чудесная	6.12e-05

The most probable tokens in the reference context with their approximated probabilities

The concept of the similar contexts

Now let's suppose we need to parse the following Russian text:

В аренду предлагается 1к квартирка, в которой есть 2к холодильник

We have 10 context positions, let's get the 100 most probable tokens for each of them.

	Context position	Probable tokens
1	В	В (0.96), На (0.04), Вашему (0.008)…
2	аренду	аренду (0.81), центре (0.06), месяц (0.02)…
3	предлагается	сдается (0.78), предлагается (0.08), сдам (0.02)…
4	1к	однокомнатная (0.18), уютная (0.11), двухкомнатная (0.06)…
5	квартирка	квартира (0.79), студия (0.05), малосемейка (0.04)…
6	в	в (0.42), на (0.12), из (0.08)…
7	которой	квартире (0.52), наличии (0.16), которой (0.02)…
8	есть	есть (0.65), имеется (0.12), стоит (0.03)…
9	2к	еще (0.3), отдельный (0.09), новый (0.07)…
10	холодильник	холодильник (0.46), котел (0.13), кровать (0.008)…

The most probable tokens in the text position contexts with approximated probabilities

The concept of the similar contexts

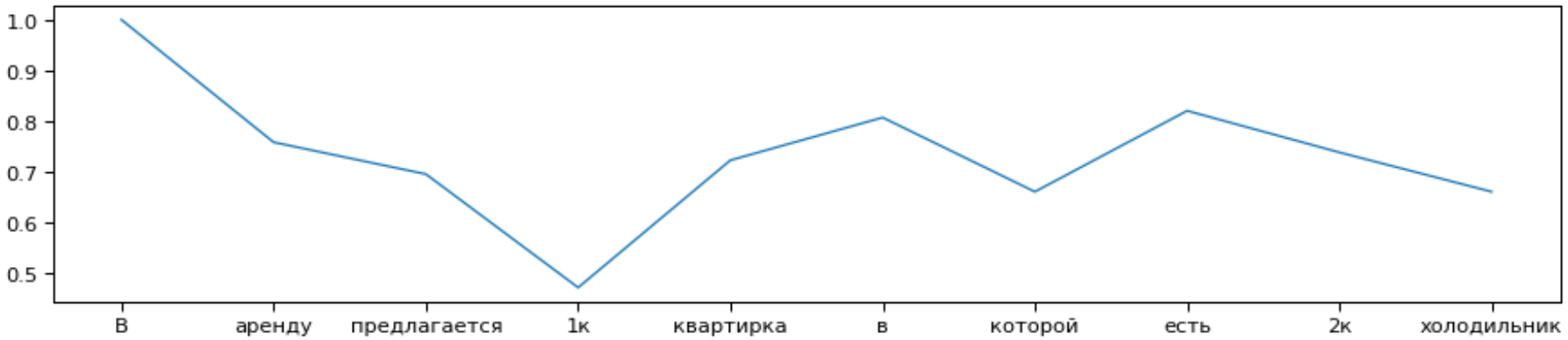
Let's calculate differences between the reference context and contexts in each text position. For this purpose, we should use a distance function. It is reasonable to try Jaro-Winkler distance since we deal with ordered (by probability) sequences.

The normalized (l2) distance difference between rival wrong "2к" and correct "1к" positions is 0.11.

	Context position	Distance	Norm. distance
1	В	0.99	0.42
2	аренду	0.76	0.32
3	предлагается	0.69	0.29
4	1к	0.47	**0.20**
5	квартирка	0.72	0.31
6	в	0.81	0.34
7	которой	0.66	0.28
8	есть	0.82	0.35
9	2к	0.74	**0.31**
10	холодильник	0.66	0.28

Approximated absolute and normalized Jaro-Winkler distances between the reference context and contexts

The concept of the similar contexts



Graph for Jaro-Winkler distances between the reference context and contexts in each text position

Magic! The desired value is most likely in the position where the "1к" token is located. The position is correct, even though the analyzed text does not contain the "Сдается" word from the reference context, and "квартира" has slightly changed.

The concept of the similar contexts

Now let's complicate the task, let's assume we need to extract a price value from very fuzzy text of a rental ad in English. First of all, we need to set up a reference context:

Price *

Using the RoBERTa model trained on English rental ads from Craigslist, we get the 100 most probable tokens that can be in the position of the asterisk.

	Token	Probability
1	now	0.08
2	range	0.05
3	today	0.04
…	…	…
16	rent	0.001
…	…	…
100	factors	8.34e-06

The most probable tokens in the reference context with their approximated probabilities for English text

The concept of the similar contexts

Let's suppose we need to analyze the following English text:

Nice townhouse, 1000 sq ft 3000 monthly

This time we have 7 context positions. Let's get the 100 most probable words for them.

	Context position	Probable words
1	Nice	Large (0.03), Private (0.026), Furnished (0.018)…
2	townhouse	apartment (0.08), studio (0.06), house (0.05)…
3	1000	community (0.01), downtown (0.007), new (0.002)…
4	sq	sq (0.83), square (0.05), sqft (0.001)…
5	ft	ft (0.78), feet (0.08), foot (0.003)…
6	3000	plus (0.07), range (0.02), from (0.01)…
7	monthly	per (0.04), rent (0.01), month (0.008)…

The most probable tokens in the text position contexts with approximated probabilities for English text

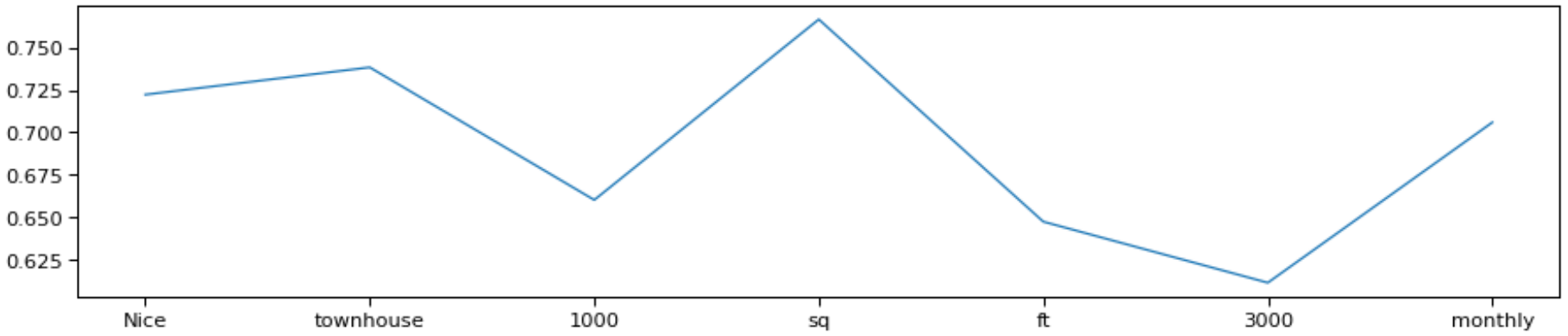
The concept of the similar contexts

By analogy with the Russian example, let's calculate the differences between reference and positions contexts

	Context position	Distance	Normalized distance
1	Nice	0.72	0.39
2	townhouse	0.74	0.40
3	1000	0.66	**0.36**
4	sq	0.77	0.42
5	ft	0.65	0.35
6	3000	0.61	**0.33**
7	monthly	0.71	0.39

Approximated absolute and normalised Jaro-Winkler distances between the reference context and contexts in each English text position

The concept of the similar contexts



Graph for Jaro-Winkler distances between the reference context and contexts in each English text position

Magic again! The minimum distance is in the correct position, where the “3000” token is located. It is despite we don’t have “price” word or its synonym.

The concept of the similar contexts

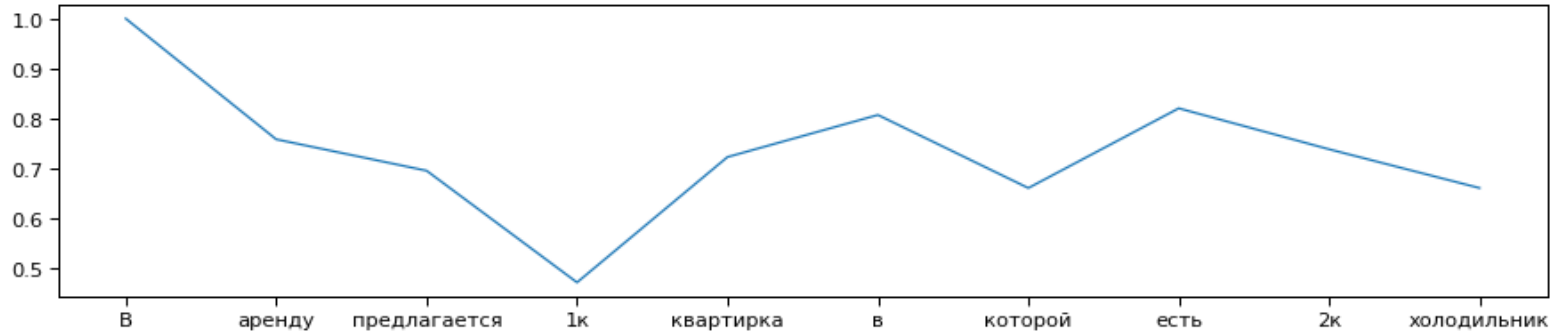
Using Jaro-Winkler distance has issue. Instead of probabilities, it uses token position, which leads to information loss. Let's implement a custom distance function that doesn't have this drawback.

Algorithm: Custom distance function

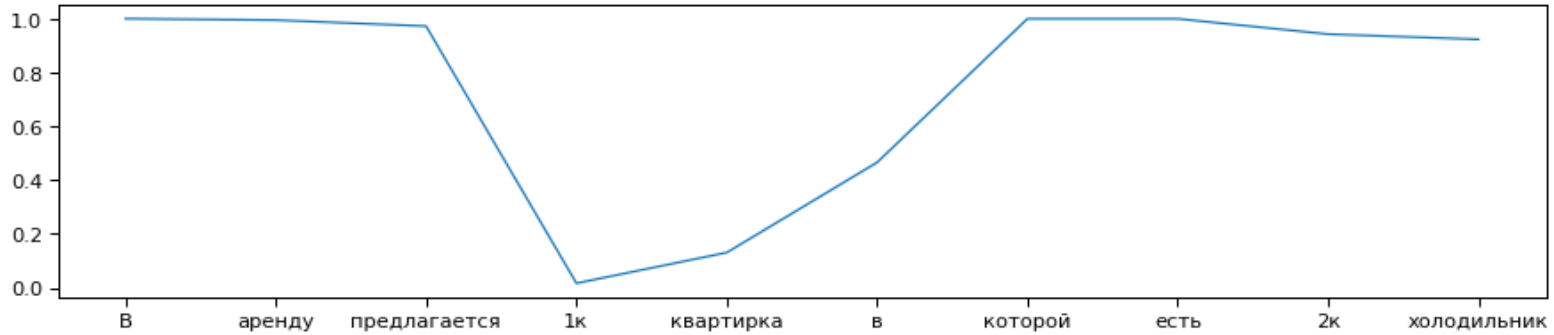
```
FUNCTION distance(reference_context, position_context):
    result ← 0
    FOR EACH item IN reference_context:
        IF item NOT IN position_context:
            result ← result + reference_context[item].prob
        ELSE:
            prob_diff ← (reference_context[item].prob - position_context[item].prob)
            result ← result + prob_diff
        END IF
    END FOR
    RETURN result
```

The concept of the similar contexts

Jaro-Winkler:

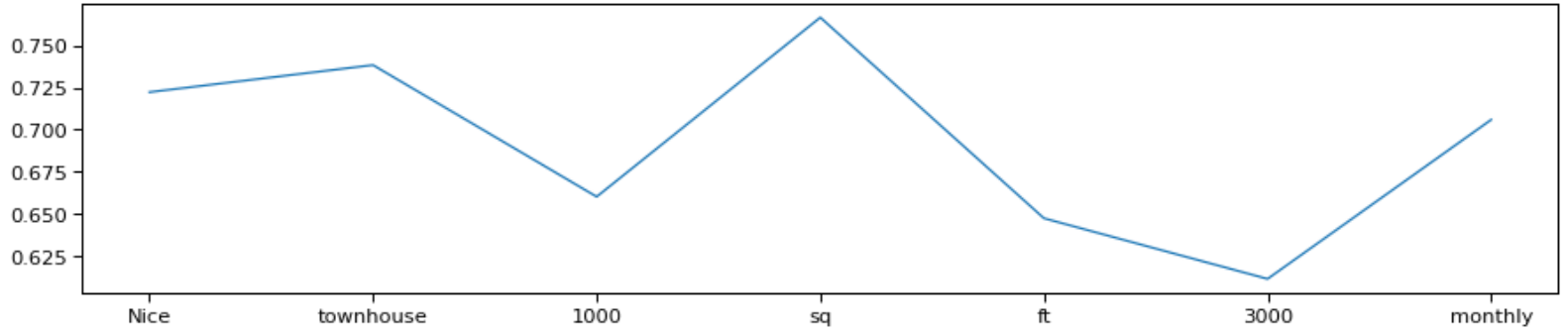


Custom:

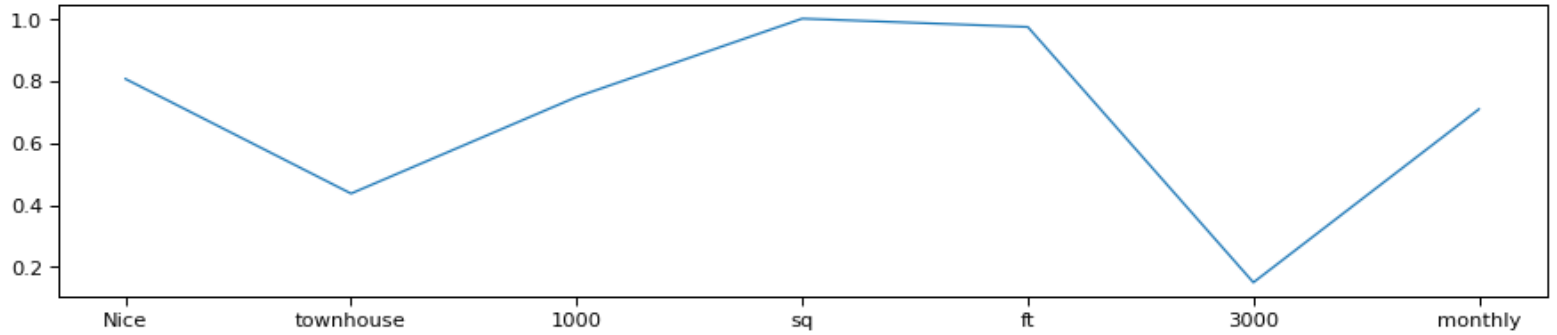


The concept of the similar contexts

Jaro-Winkler:



Custom:



According to the obtained results, the custom distance function works better. But possibly for other languages, domains, or datasets another function may be better suited. For this reason, it was decided not to hardcode it but to make it possible to define it by the framework parameter.

Testing the hypothesis of the similar contexts

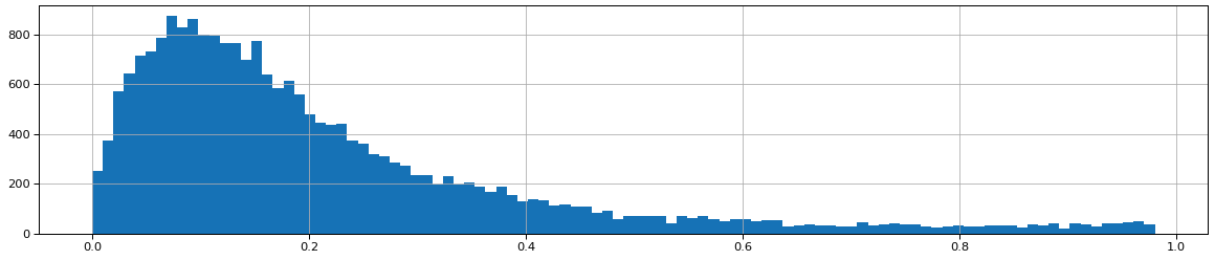
To prove he hypothesis, we should rely on statistical hypothesis tests.

Let's analyze the distributions of distances to the reference context for right and wrong positions.

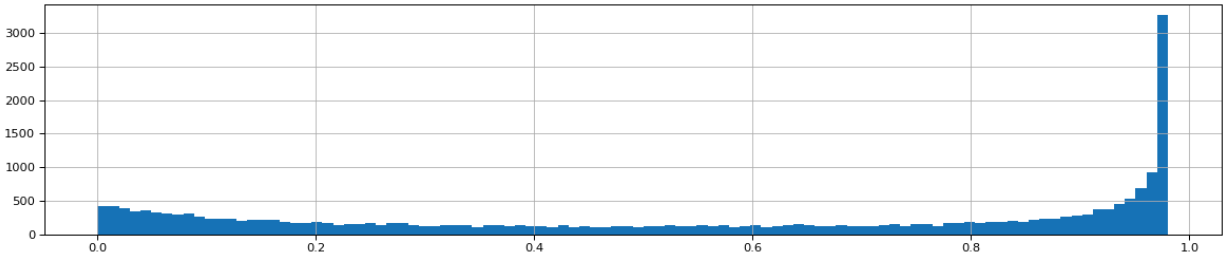
The distribution of distances on the graphs shows that they are probably not normally distributed, given their high asymmetry (skewness).

The Shapiro-Wilk test, as expected, gives a p-value ($\approx 3^{-25}$) considerably less than the 0.05 threshold for both sets, which confirms that the distributions are not normal.

Distances for right positions:



Distances for wrong positions:



Testing the hypothesis of the similar contexts

To prove the concept, we should confirm that distances in right positions are less than in wrong ones. So we need to reject the opposite meaning.

Since we have not normal distributions, we should use the Mann-Whitney U test.

Running a one-tailed test gives us a p-value ($\approx 2.55^{-64}$) considerably less than the 0.05 threshold.

So we reject the null hypothesis in favor of the alternative one. Which means that the hypothesis of the similar contexts has been **proved**.

$H_0 : \mu_{wrong} \leq \mu_{right}$ *(distances in wrong positions are equal or less than in right ones)*

$H_1 : \mu_{wrong} > \mu_{right}$ *(distances in wrong positions are more than in right ones)*

There are two possible outcomes: if the result confirms the hypothesis, then you've made a measurement. If the result is contrary to the hypothesis, then you've made a discovery.

Enrico Fermi

Testing the hypothesis of the similar contexts

We do not live in an ideal world, so collisions are possible. Such impressive results were achieved using state-of-the-art RoBERTa models that trained on a powerful GPU for **several weeks**.

If the neural network is poorly trained the number of errors may increase. Therefore, the similar contexts approach is not a silver bullet, it is not reasonable to rely solely on it.

It is better to use a combination of several approaches. The FuzzyText framework follows this tactic.

We do not live in an ideal world :(

FuzzyText framework installation

The FuzzyText framework is implemented as a Python package.

The latest version is available at its GitHub repository. It is also uploaded to the Python Package Index (PyPI).

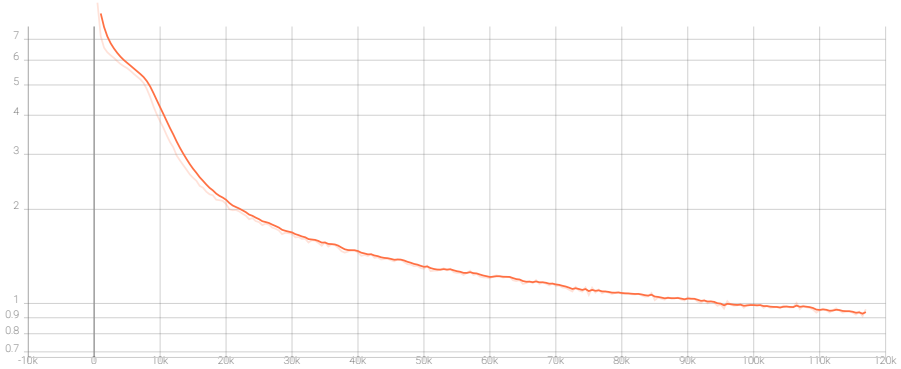
pip install fuzzytext

Language models compatible with the FuzzyText framework

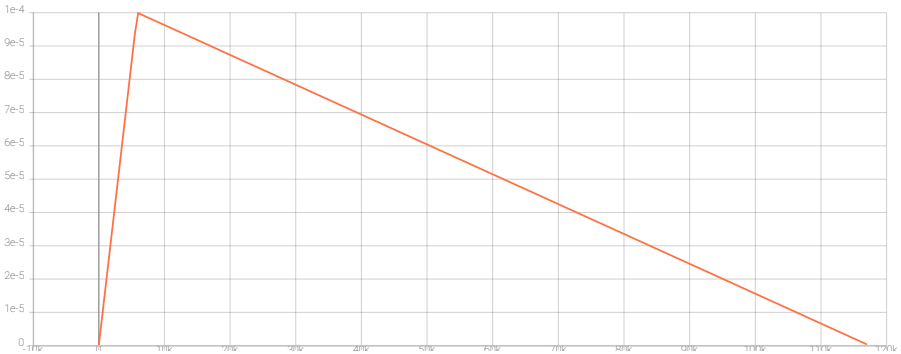
The framework supports all the compatible models with Hugging Face Transformers, including ALBERT, BERT, DistilBERT, GPT, and RoBERTa. Also, it supports N-grams language models.

It is recommended to use models based on a neural network for this framework. However, it is essential to train such models correctly.

The tools for training models can be found in the GitHub repository of the framework.



Loss decreasing of the RoBERTa training process using the Russian estate renting dataset



Learning rate warm-up for the RoBERTa training process using the Russian estate renting dataset

Basic examples of the FuzzyText framework usage

Let us assume we need to extract the number of rooms from a fuzzy Russian text. The Python code on the right is a basic approach to do that. It returns the following result:

```
{'pos': 3, ‘value': '1к', 'score': 0.9944143216925971}
```

```
from fuzzytext import LanguageModel, Extractor

language_model = LanguageModel(
   model_type="transformers",
   model_path="rafagudinov/ru_rent_estate_ads"
)

rooms_extractor = Extractor(
   language_model=language_model,
   reference_context="Сдается * квартира",
)

rooms_extractor.extract(
   "В  аренду предлагается 1к квартира”,
   "в которой есть 2к холодильник"
)
```

Basic examples of the FuzzyText framework usage

This time let us assume we need to extract the price from a fuzzy text. The code on the right gives the following result:

```
{'pos': 5, 'value': '30000', 'score': 0.7797146228563306}
```

There are several framework configuration parameters to fine-tune it for custom needs. More details can be found in the GitHub repository.

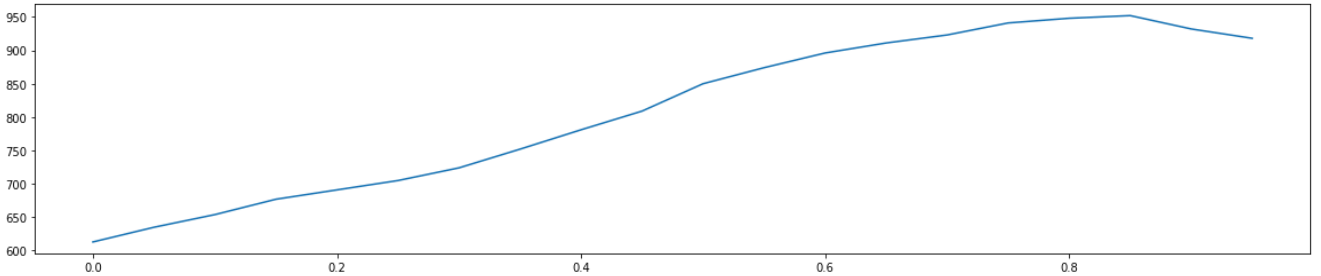
```
from fuzzytext import LanguageModel, Extractor

language_model = LanguageModel(
    model_type="ngrams",
    model_path="models/ngrams/ru_rent_estate_ads"
)

price_extractor = Extractor(
    language_model=language_model,
    reference_context="Цена *",
)

price_extractor.extract(
    "На Ленина 15 сдается квартира, 30000 + ку",
    "звоните: 8(495)1234567"
)
```

Similarity vs. position dualism

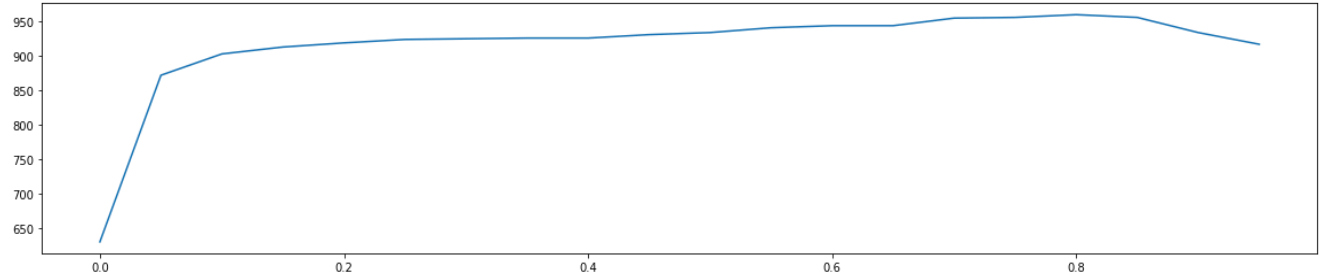
The score of extraction is calculated based on the similarity and position scores. For estate rental domain the best result if the rain between them is about 0.8.

For other languages and domains it may have another optimal value. So it is a framework hyper-parameter (sp_ratio).

Accuracy for N-grams based extractor



Accuracy for RoBERTa based extractor



Metrics of the FuzzyText framework

It is a non-trivial to set up a baseline since advanced state-of-the-art solutions are proprietary and not publicly available. So, we will use a basic solution based on the edit distance.

People usually make mistakes in a middle of a word, it is better to use the Jaro–Winkler distance instead of Levenshtein.

Even out-of-the-box framework gives **better** result than the baseline.

Method	F1-score
Baseline	0.88
Framework (N-grams)	**0.94**
Framework (RoBERTa)	**0.96**

Datasets

While working on this project, two datasets were fetched for testings and measuring metrics. Both are real estate rental ads.

The first one is in **Russian**. It is retrieved using the official VK API. It contains 4M+ unique ads. Based on this dataset, a derivative one was created in which the values of room numbers and their positions were labeled.

By analogy, **English** dataset was created. It is fetched from Craigslist using Scrapy. It has 200K+ unique ads.

Future work

To make this framework the state-of-the-art solution for structured data extraction from fuzzy texts.

It should be as extensible as possible, so other developers can write plugins.

Promote the concept of similar contexts and the framework through writing articles (Medium, Habr) and speaking at professional conferences.

Conclusion

In this project, the key hypothesis that **similar contexts tend to have the same words** has been raised and confirmed. It can be considered a breakthrough.

Based on this, the core module of the framework (FuzzyText) has been implemented. Even out-of-the-box, it gives excellent results. This should save the nerves of many software developers.

The idea of working on this project arose out of the need to extract structured parameters from very fuzzy Russian real estate rental ads. The popular approaches have issues, so a new tool was required. As a result of this project, such a tool has been created.

Links

- https://github.com/ralan/fuzzytext - the FuzzyText framework source code
- https://github.com/ralan/HSE-project - Jupyter notebooks for similar contexts hypothesis proof, and FuzzyText framework experiments
- https://github.com/ralan/fuzzytext_models - n-grams models
- https://huggingface.co/rafagudinov - RoBERTa models
- https://www.linkedin.com/in/rafagudinov - LinkedIn profile

Acknowledgments

I put much effort into solving the problem using clustering. As a result, I have gained a much deeper knowledge of this subject. So I would like to thank my thesis supervisor HSE Professor **Vasilii Gromov**, for advice to try the clustering approach. Also, I am grateful for the suggestion to implement the N-grams method in addition to Transformer based one. In some cases, N-grams may be preferable.

I want to express my gratitude to **Roman Pavlushko** (Ex-CTO Avito.ru, Highload program committee member) for his consulting and the opportunity to implement and test the framework in an active business project other than the real estate market.

I am grateful to HSE Associate Professor **Ilya Schurov** for his consulting and good advices.

I want to thank the **creators** of the HSE Master of Data Science Online Program. In this project, I used a lot of knowledge I gained in the learning process.

And definitely, I am grateful to my classmates, friends, and family. I dedicate this project to **my mother**.