DMITRY CHAPLINSKY

PYTHON SUPERPOWERS ON CIVIC
SOCIETY'S SECRET SERVICE

1. Is it the same talk as I gave on PyConBy? No, this one’s completely different — it's in English. So you are going to suffer a little bit.

2. Talk format. I'm not here to teach you anything new or to show unusual aspects of well-known stuff. My talk is more like a report of what we've been doing for the last two years.

First of all, I'm here to tell you a story, with all the ups and downs. My goal is to inspire you and to show you that IT technologies, along with volunteers and a great will can bring about a change.

Introduction.

01

THE BEGINING

< 2 >

The history of White Collar Hundred, an NGO

FEBRUARY 2014 — ...

Chapter 1: History and the beginning of the Unshred project.

We begin with the White Collar Hundred, an NGO.

It all began at the end of the Maidan.

When Mr. Yanukovych left the building, journalists and activists found tons of documents. In just a few days the pile got even larger: dozens of trash bags with documents were found near the office of a Ukrainian oligarch Kurchenko, the runaway.

Sergey Kurchenko

29 y/o. Multi-millionaire.

And when I say paperwork or documents you might think of orderly shelves with endless stacks of folders. Unfortunately, that wasn’t the case. The documents the activists got were either half-burnt, or soggy, or torn, or shredded.

Not everything was lost, though. Apparently, if you have a room big enough and enough volunteers (a LOT of volunteers) you can make substantial progress with such documents. It's not that hard to reconstruct torn documents. It's harder to do the same with documents that were cut into strips: it takes more time and a pair of tweezers. Also, it hurts your back, your eyes and you'll end up with a flat butt. And, this is how the White Collar Hundred was born.

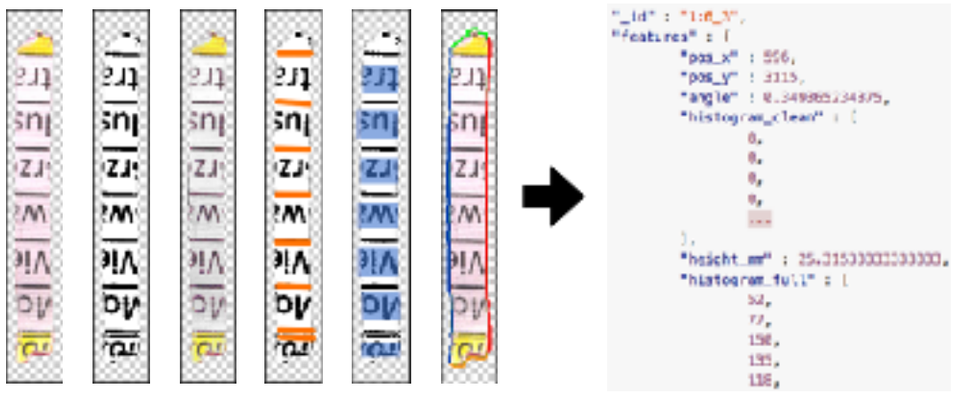
But what about the shredded documents, you might ask?

«The Documents»

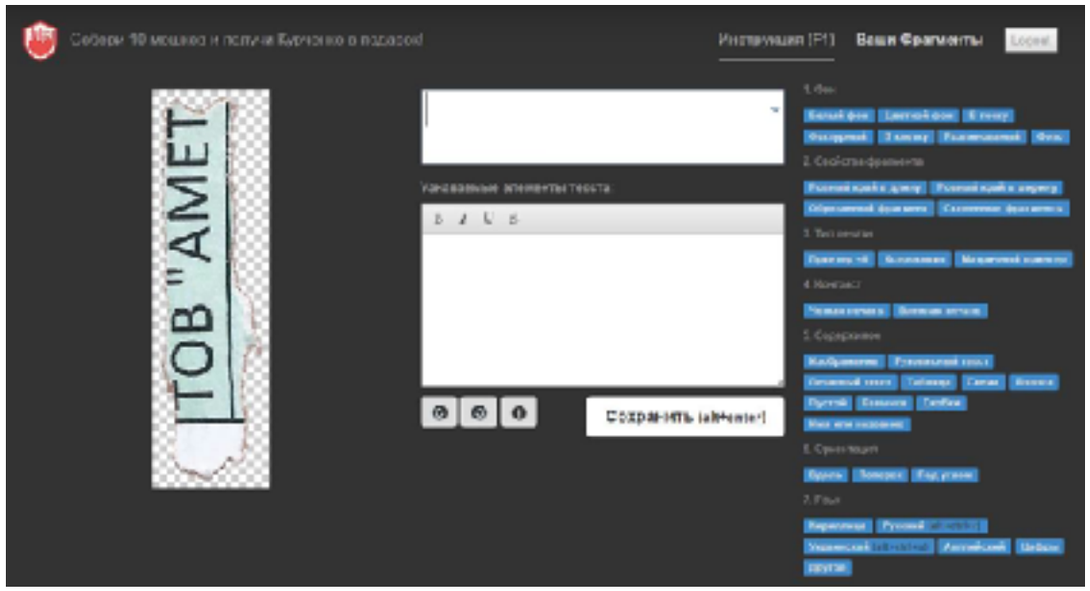
We've took an intuitive path towards manual reconstruction of documents. In order to reconstruct a page, you must start with something, a crystallization seed of some kind. Much like with a jigsaw puzzle, when you begin with corners, or edges, or faces. For a document, it might be a header, or a fragment of a stamp, etc. But how can one find a shred with such distinctive features if one has gazillions of them?

Together with Denys we worked out the following workflow to speed up the reconstruction process, using latest IT technologies:

unshred.it



1. First, shreds are digitized. That is, they're glued on colored paper and scanned in. Luckily, that was already done.
2. Then each picture must be processed: shreds must be detected on the sheet, extracted and normalized.
3. Additional features should be calculated for each shred: geometry, palette, etc.
4. Some attempts might be made to calculate higher-level features: such as "has_lines", "is_a_photo", "has_pen_marks", etc.



5. Then pre-processed shreds are loaded into a crowdsourcing platform, which uses volunteers to extract even more information from each shred. This way chunks of text and details of paper quality, text direction, font families, etc. are obtained.
6. Finally, data from shreds is used to calculate various similarity metrics to narrow down the search.

With the help of my friends I've written two components of the system:
1. A visual recognition component, which detects shreds from the colored page background, cuts them out and calculates all kinds of fancy features.
2. A crowdsourcing platform that allows task upload and then processes those tasks with the help from volunteers.

Technology stack

CV part

- Python
- OpenCV
- PIL

Web part

- Python
- Flask
- Mongo
- jQuery

02

OPEN
DATA. NOT

Asset declarations of Ukrainian officials

DECEMBER OF 2014—…

The next chapter is on assets declarations of Ukrainian officials.

As some of you might know (and most of you clearly don't), a lot of Ukrainian officials are obliged to fill out an annual asset declaration. Those declarations must then be published on the state administration website in electronic form. Here's the problem: we have different views on what constitutes "electronic form."

E-form of asset declaration

Needless to say, governmental bodies usually neglected to publish such declarations and whan they did, they were hidden in dusty folders in a paper form.

The lustration [vetting?] process uncovered a lot of these dusty documents and our friends from lustration committee shared those scans with us.

What can we do with about 3000 paper scans, some of each are hardly legible? We decided to digitize them with the help of volunteers. So, to progress faster we've constructed a monstrous Google form that had the fields of the paper declaration. Then we asked volunteers for help. And, within two days (right before the Orthodox Christmas) every declaration had been deciphered (with a

Vulyk: crowdsourcing framework

I've taken the core of crowdsourcing part of Unshred and with the help of my friends (Hi Dima and Vova!) turned it into the framework. An Amazon Mechanical Turk of our own that takes a burden of task management, volunteer registration and can actually assign to volunteers exactly as many tasks as they are capable of doing. This is how the Vulyk project has started.

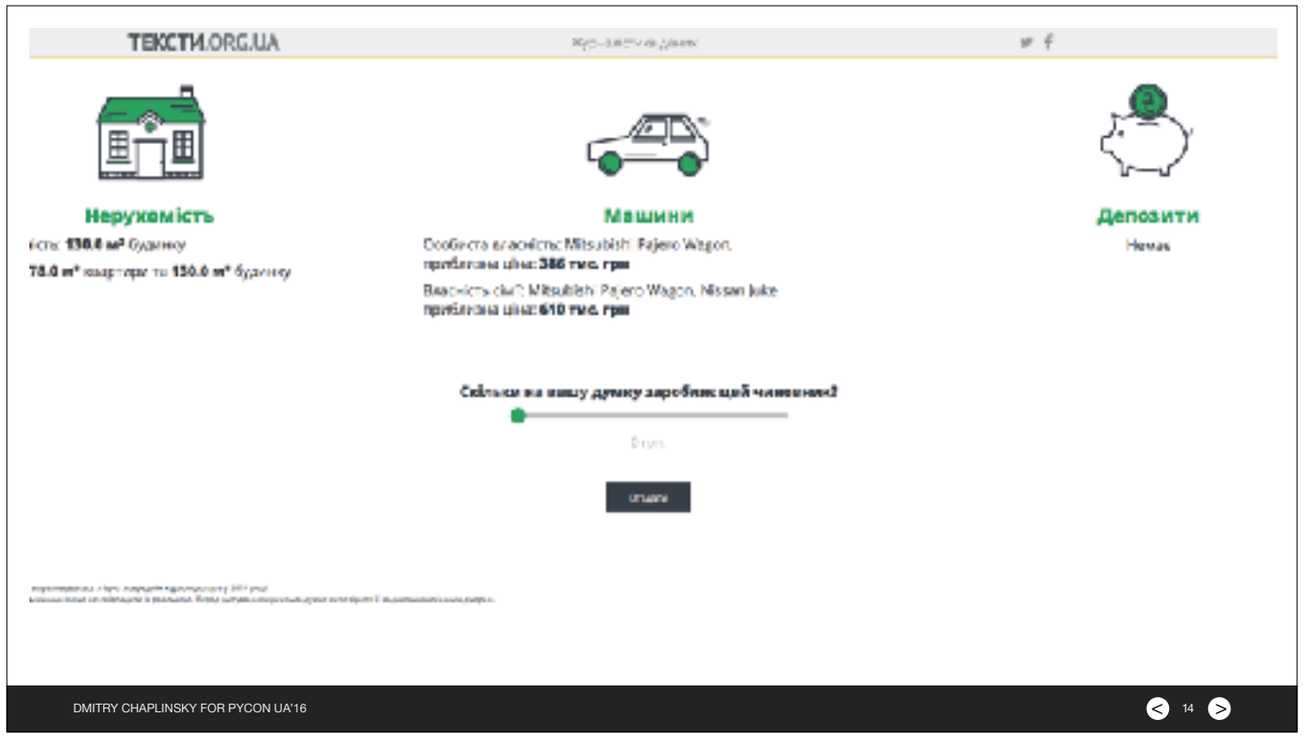
Then another friend of mine, also Dima, made a huge contribution and helped us create a digital rendition of the declaration form. We tried our best to get rid of issues we had with the lame Google Form: we added validations, hints, autocomplete fields and broke it into sections. Then we wrapped it into the very first plugin for the Vulyk. Loaded some tasks there and invited volunteers.

In parallel with that, we've made a website where one can browse digitized declarations. We've added a proper search engine, so now it's much easier to find all declarants with a Mercedes, for example. We've published the first batch from the Google form there. And, soon enough we've began adding data digitized via the Vulyk.

And the most important…

Now it’s an **OPEN DATA**

Also, we gave an API access to all the data we had. That paid off immediately. In (literally!) just a few days a guy from Dnipropetrovsk appeared, who published analytics on top of our data. He sliced and diced that data in every possible way. He was also kind enough to share an R script to generate the article. So we've integrated that program into the site, so now our analytics page is always generated from fresh data.



And that wasn't the only case. Our partners from texty.org.ua made a simple yet powerful BI, which allows analyzing data interactively. Finally, they made a neat browser game out of it! They are showing players the assets listed in the declaration and then let them guess the annual income! It's fun to play!

This is, by the way, one of the signs of a successful project— when something new is being created out of it. I've only mentioned software products, but there were also numerous investigations and articles based on the data we've digitized.

Workflow

- Find and collect scans from the public websites…
- …or obtain them via RPI
- Pre-processing of the source files
- Digitization by volunteers (each declaration is being digitized by at least 3 different people)
- Post-processing of the data
- Editing of the data
- Publishing
- Updating of the analytic

Since the beginning of the project, we've published about 20k of declarations online. Now it's a well-organized process which looks like this:
1. We search for scanned declarations using specially defined criteria.
2. Then we normalize source files and convert them into a PDF.
3. Those declarations are digitized by at least three different volunteers.
4. Then we post-process results using dozens of rules: to fix common mistakes, strip extra spaces, etc. Processed data is then saved to a giant Excel spreadsheet.
5. Then it's handled by an editor, who uses specially written VBA macros, which allow quickly jumping between discrepancies in the data and fixing them.
6. Then we publish the data and update the analytics.

Technology stack

Collection & digitization

- Python
- Flask
- Mongo
- Vulyk!
- NLTK
- OpenRefine
- Excel & VBA

Web part

- Python3
- Django
- Jinja
- ElasticSearch
- PostgreSQL
- R

We'll return to this project later. Now I want to present to you another one. Another chapter in the book. The PEP project.

03

NOT THE PEPS
YOU'VE USED TO

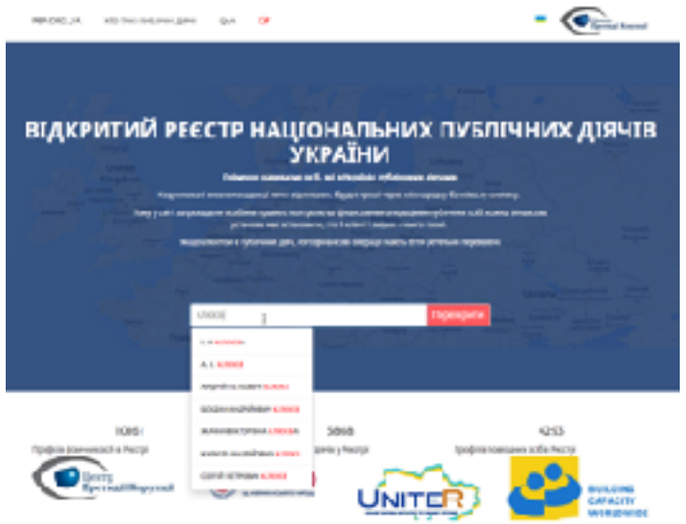
Politically Exposed Persons

DECEMBER OF 2014—…

Despite the fact the PEP abbreviation might be familiar to you, it's a totally different thing. PEP is the term from the world of financial monitoring. PEP stands for Politically Exposed Person. It’s either a politician, or his close associate: a business partner, a member of the family, etc. Why does it matter? Because for those guys, banks have different rules for financial monitoring. It's harder to receive a loan, transactions of PEPs must be monitored more thoroughly, etc. The only problem is that banks don't generally know if the person is a PEP. Of course they ask about it on all of their forms when opening new accounts, but honesty is rarely a virtue among politicians.

Of course, there are some international registries from private companies. Banks are paying for access and can run checks against these databases. This solution

pep.org.ua: Open registry for financial monitoring needs



With that in mind, the AntAC (a prominent Ukrainian NGO) decided to build their own registry. So they've partnered with us. They began collecting information using requests for public information as well as MP's requests.

Those responses are then scanned and digitized into a Google spreadsheet. Why Google Spreadsheet, you'll ask? Because it's a very familiar interface for people who digitize the replies. It also allows collaborative work and can be easily imported into the site on a daily basis.

- To check quickly, if the person is a PEP
- To obtain the document that verifies that fact
- To see his bio, career and connections
- In Ukrainian and English

After that, the imported data is translated, improved and indexed by the search engine. The general goal of the site is to answer quickly whether or not the given person is a PEP. In addition to that analysts and investigators of the AntAC are collecting and adding extra information to PEP profiles. It might be textual information, a dossier, or information from media or connections. Connections to other persons, companies and countries. So we are trying to provide as much of supporting information about the person as we can.

The site is available in two languages: Ukrainian and English. The issue of different romanizations of names is also addressed.

Korolevs′ka Natalìâ Ûrïivna
Corolevs'ca Nataliia Yuriivna
Korolevs'ka Natalija Jurijivna
Korolevskaja Natal'ja Jurievna
Korolevskaia Natal'ia Iur'yevna
Korolevs'ka Natalija Juriyivna
Korolevskaja Natal'ja Jur'evna
Korolevskaya Natal'ya Yurievna
Korolevska Nataliia Iuriivna
Королевская Наталия Юрьевна
Korolevs'ka Nataliia Yuriivna
Korolevska Nataliya Yuriyivna
Korolevskaya Nataliya Yur'yevna
Korolevskaia Natalia Iurievna
Korolevska Nataliia Yuriivna
Korolevskaja Natalija Jur'evna
Korolevskaya Nataliya Yurievna
Korolevs'ka Nataliya Yuriyivna
Korolevs'ka Natalija Juriivna
Korolevska Nataliia Iourïivna
Korolewska Natalija Jurijiwna
Korolevskaia Natalia Iurevna
Королевская Наталья Юрьевна
Korolevskaia Natal'ia Iurievna
Korolevskaja Natalija Jurievna
Korolevskaia Nataliia Iurevna
Korolevskaia Nataliia Iurievna
Королевская Наталья Юриевна
Korolevs′ka Natalija Jurijivna
Korolevskaya Natal'ya Yur'yevna
Korolevskaia Nataliia Iur'yevna
Королевская Наталия Юриевна

Let me explain this a little bit.

Slavic names have all different kinds of romanization or transliteration. I know 3 or 5 versions of my name. Now imagine an average bank employee from Cyprus. He sits and checks names in transactions against the database. The difference in just one character will allow him to say: hey, these are two completely different people. That's why we are translating all names also into Russian and then translitering them using 19 different schemes. On the slide you can see the results of this algorithm used on the name of one such official.

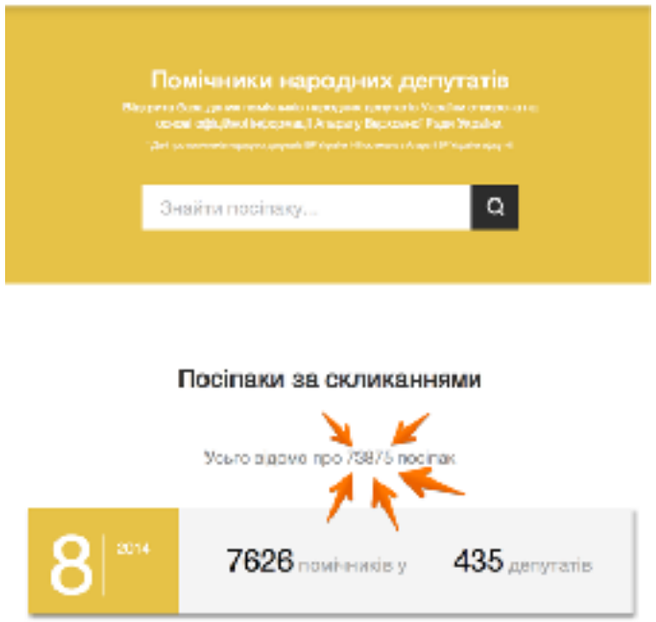
Technology stack

- Python
- Django
- Jinja
- PostgreSQL
- ElasticSearch
- WagtailCMS
- Django Model Translations

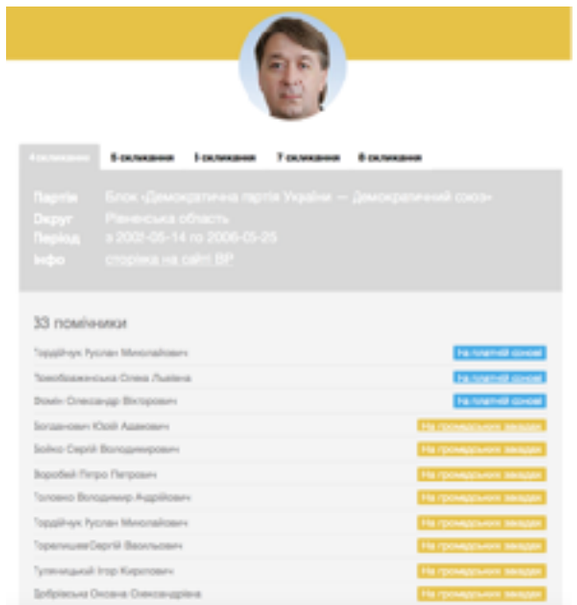
Right now the site knows about 6300+ of PEPs and 5600+ close associates. One might ask, how did we find the information about close associates, such as family members? It's easy. They're mentioned in declarations. We've matched two datasets, PEPs, and declarations with the help of a fuzzy search. Two volunteers then checked the results. That allowed us to add not only family members but also information about assets and incomes. Exactly what banks want to know! Neat!

Ok, now the next chapter. Quite a funny one.
Posipaky.

Open registry of assistants of Ukrainian MPs



Investigative journalists from Slidstvo.info obtained an interesting dataset from the Parlament. A list of assistants of Ukrainian MPs for the last five convocations. It's okay to be skeptical here, but it's huge and exhaustive. It has more than 70k records. Why is it important? Because assistants of MP’s are often playing a serious role in corruption schemes. Knowing that some person was a minion of an MP gives us a cinnection. For some investigations, that is a missing link.



- Information from official source (The Parliament)
- 5 convocations (14 years)
- 2400+ MPs
- 73000+ assistants
- Full-text search
- Different points-of-view on the same data: by convocation, MP, assistant

The data came in a Word document format. I wrote a simple parser, massaged the data a bit and linked it with the convocation lists from the Parlament website and Wikipedia. We've also reconstructed connections, so it became visible how different assistants changed their masters over time. This added a whole new dimension to the data and led to interesting discoveries. What I love about owning the whole dataset is that you can always find something new. Facts that weren’t clearly visible before.

Technology stack

- Python3
- Django
- Jinja
- PostgreSQL
- ElasticSearch

Oh and something else to mention, the name. Posipaky is Ukrainian for “minions”. And it caused a proper butthurt for some MP's and their assistants. MP Hanna Hopko even mentioned the project from the tribune of the Parlament. That was quite a rewarding moment indeed.

05

KIND OF OPEN DATA

< 26 >

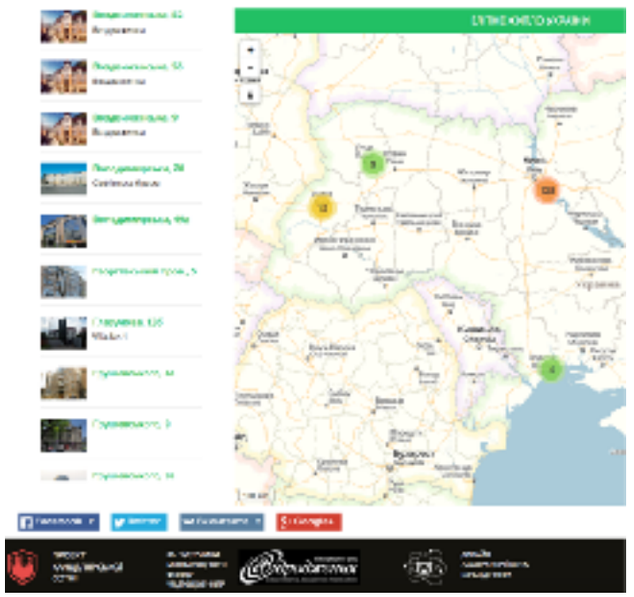
Other projects, related to the open data

JANUARY OF 2015—...

In addition to that, we've also done a few smaller projects on open data, one of which I want to describe briefly.

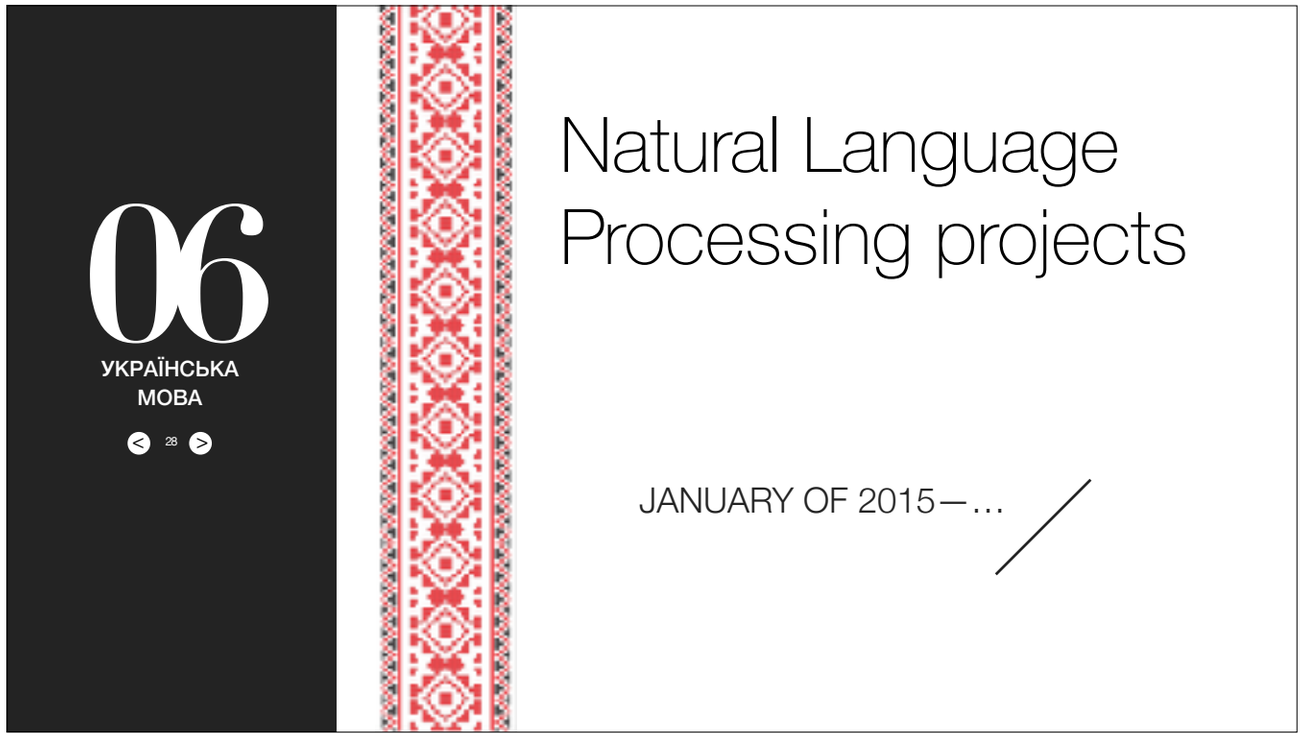
GarnaHata. A registry of owners of luxury apartments.

GarnaHata: registry of the owners of elite real estate



The Ministry of Justice maintains a registry of real estate owners. It was first openen to the public in the autumn of 2014. It was a huge success at that time, but the registry lacked some important features. First of all, you were only able to search by address. Also, it wasn't free. Also, information from the registry was returned in a form of a PDF document.

We decided to fix that. We obtained a lot of extractions from this registry. Not random ones, no. We targeted luxury real estate all across Ukraine. We've learned how to parse those PDF's files. Then we processed the parsed data to strip it of sensitive data, such as apartment numbers. And we created a website with all that information. We put it on a map; we added a proper search engine; we disclosed all that information in an API as well. So it was now possible to search

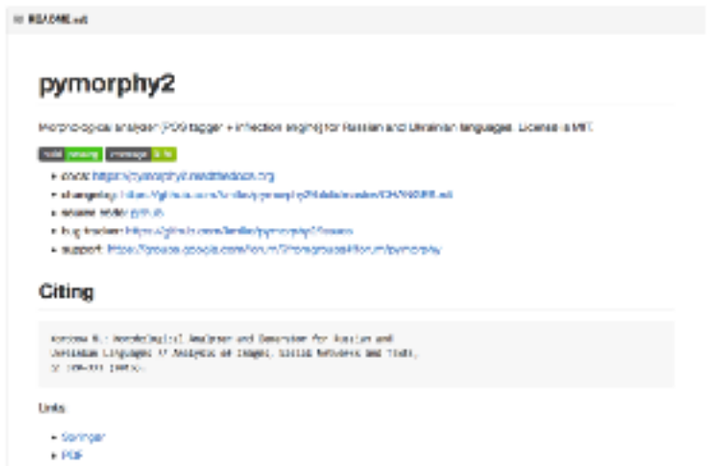


Okay, enough of open data and transparency initiatives.

Let's talk a little bit about Natural Language Processing.

I love NLP. Unfortunately, for the Ukrainian language, it's still in an embryonic state. We decided to fix that. For that goal in mind, we collaborated with a great team of linguists who are working on an open morphological dictionary and an open corpus of the modern Ukrainian language.

Support for Ukrainian language in PyMorphy2


README.md

pymorphy2

Morphological analyzer (POS tagger + inflection engine) for Russian and Ukrainian languages. License is MIT.

build passing coverage 91%

- docs: http://pymorphy2.readthedocs.org
- stackoverflow: http://stackoverflow.com/questions/tagged/pymorphy2
- release info: pypi.org
- bug tracker: https://github.com/kmike/pymorphy2/issues
- support: https://groups.google.com/forum/?fromgroups#!forum/pymorphy

Citing

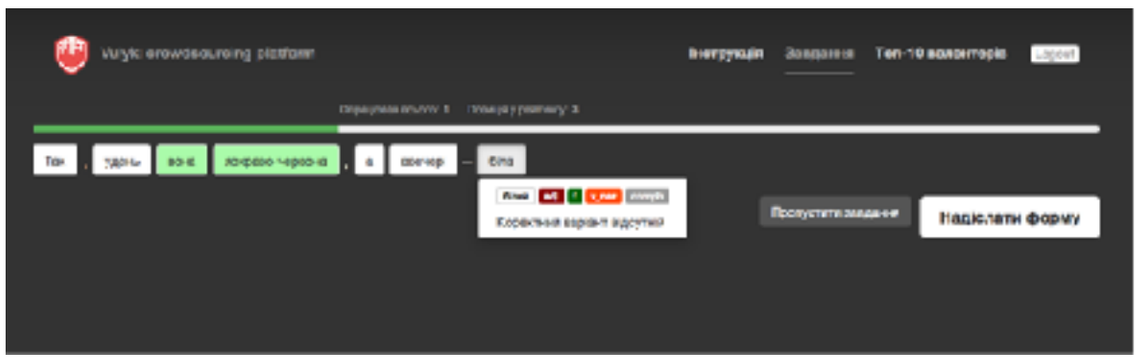
```
Korobov M. Morphological Analyzer and Generator for Russian and
Ukrainian Languages // Analysis of Images, Social Networks and Texts,
2015.
```

Links

- Springer
- PDF

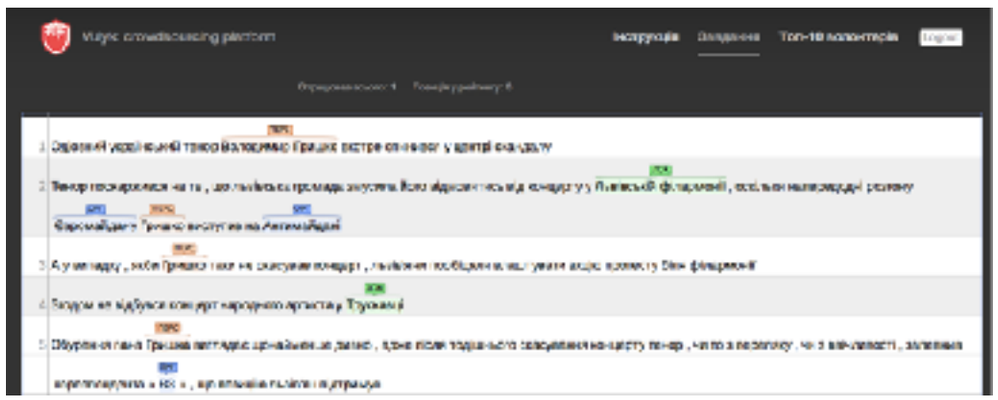
First of all, with the help of Michail Korobov we've added a Ukrainian language support to the PyMorphy2, an open source morphological analyzer.

Project to create a very first open corpus of Ukrainian language



Then we created two more plugins for the Vulyk. One allows a team of linguists to tag texts from the corpus with Part-Of-Speech tags in a quick and efficient way.

Project to tag NER entities in that corpus



The second plugin is created to tag the named entities (names, organizations, geopolitical entities, etc.) in the same corpus. Why do we need a tagged corpus? Well, first of all, it's fun to create an open corpus that everyone might use for their NLP tasks. Also, we need Named Entity Recognition for our upcoming projects.

Support of Ukrainian language in ElasticSearch

Oh! And here’s one more project I promised to mention!

For that one, I built a dictionary, and Dima Hambal wrote the plugin for our search system of choice: the ElasticSearch. So now it's possible to perform a full-text search in Ukrainian using ElasticSearch. That allows searches using any word form, singular or plural, inflected or not. Yay!

07

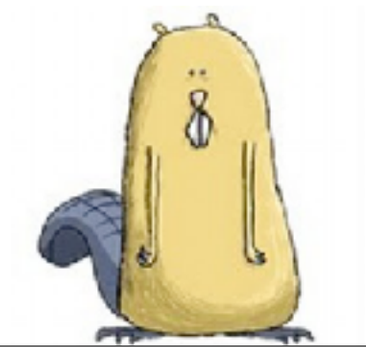
MANY THANKS TO

Denys Bigus
Dmitry Hambal
Artem Hluvchynskyi
Dmitry Nechipurenko
Andrey Turik
Volodymyr Hotsyk
Markiyan Yurynets
Olha Makarova
Oleksandr Chaplinsky
Andrey Medvedenko
Oleksandr Botezatu
Anatoly Bondarenko
Kyryl Zacharov
Sergey Vorontsov
Stas
Oleksandra Dubicheva
Khina Chlek
Tata Peklun
Andrew Shuran
Andrew Rysin
Mariana Romanyshyn
Sasha Drik
Vsevolod Solovied
Mikhail Korobov
and yet another 3-4
thousands of people

08

WEIRDOBEAVER

But why? WHY?!



Now you might have a question. Why? Why collect all that data about entities and connections, learn how to analyze raw texts, etc., etc.

I won't answer this just yet.

Instead, I'll show you a glimpse of our upcoming project. It doesn't have a proper name yet, but it's cool.

09

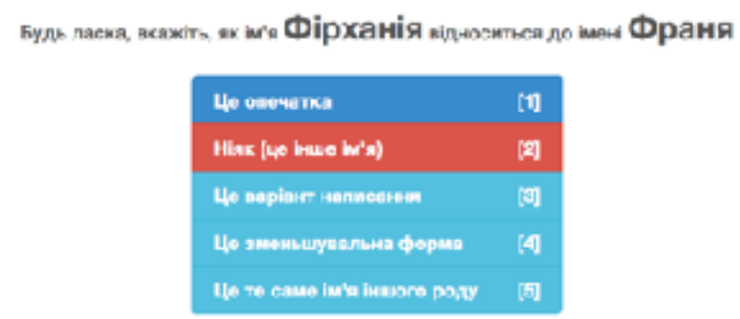
IN THE NAME OF

NameS. With capital S

FEBRUARY OF 2016—…

It's all about names. We collected quite a lot of datasets on names. And with names I mean not only first names but also patronymics and surnames as well. Then we tried to translate them into all popular languages as well as transliterate them using all of the different schemes I mentioned above.

Quite a lot of datasets on names



We've also collected and manually processed a list of possible typos for those names and their diminutives (using Vyluk of course). And with the help of PyMorphy2 we've learned how to inflect those names in Ukrainian and Russian.

That we then put into the graph DB

Finally, we've loaded them into the Neo4J, a graph database.

What for?

- To be able to compare and match persons from the different datasets…
- …even if they are noisy and in different languages
- To generate all possible spellings and inflections of the name for the search indexer
- And to do a naïve named entity recognition. Language-agnostic one!

What for, you might ask? To solve the following tasks:
1. Quickly matching persons from different datasets that came from different sources, no matter which language those data sources are in or how noisy they are. For example, we might match PEP with persons mentioned in Panama Papers (no, we can't yet, as Panama papers are not publicly available, but you get the idea). Also, because we know how names and patronymics are connected we might also find potential relatives in those datasets. Say, brothers/sisters. Or sons/daughters/parents.
2. We can use it for naïve but powerful NER and analyze raw texts, no matter which language they are written in.
3. Finally, we might generate all possible combinations of the name spellings to create a better search engine for projects like PEP.

https://github.com/dchaplinsky/

That's all for today, folks, hope to see you next year :)