Timeline

Project: Greek Government Gazette Text Mining, Cross-Linking and Codification - 3gm

GitHub Repository

About

This is the timeline kept during GSoC 2018 for the 3gm project. It contains a summary of commits done during each week along with TODOs. The wiki of the project is kept here.

Weeks

Week 1 (30/4 - 6/5)

1. Repository Creation
2. Abstract Upload
3. Dataset Download (~50 public documents in PDF format)
4. Parser Script Created
    - Separation into Articles
    - Extraction of Dates
    - Document Cleanup from punctuation and hyphenation
    - Extraction of extracts (things that would be added to laws) and non-extracts (possible commands such as additions, deletions, amendments etc.)
    - Word2Vec Model Trained on raw articles
    - Dimensionality Reduction on Word2Vec vector file with t-distributed Stochastic Neighbor Embedding.
    - Extraction of Signatories
5. Simple Classifier of Actions based on Levenshtein Distance & other heuristics
6. Dataset Visualization

TODOs for Week 2 (7/5 - 14/5)

1. ~~Continuous Integration with Travis~~
2. ~~Code refactoring~~
3. ~~Database Setup (MongoDB)~~
4. ~~Improvements on heuristic search~~
5. Read on guides on codification
6. Parse Government Gazette Issues which are stored in HTML (Not a good option at all)
7. (Optional) Study Greek Universal Dependencies https://github.com/papachristoumarios/UD_Greek-GDT and how they can be used.

Week 2 (7/5 - 14/5)

1. Improvements on heuristic search.
2. Documenting existing code
3. Writing the first unit tests (for simple examples)
4. Study existing literature (mainly on ISOKRATIS legal database)
5. Continuous Integration with Travis CI
6. MongoDB integration
7. Read on guides and literature on codification (mainly already codified documents)
8. Experimenting with Latent Dirichlet Allocation for topic modelling (either on articles themselves or issues).

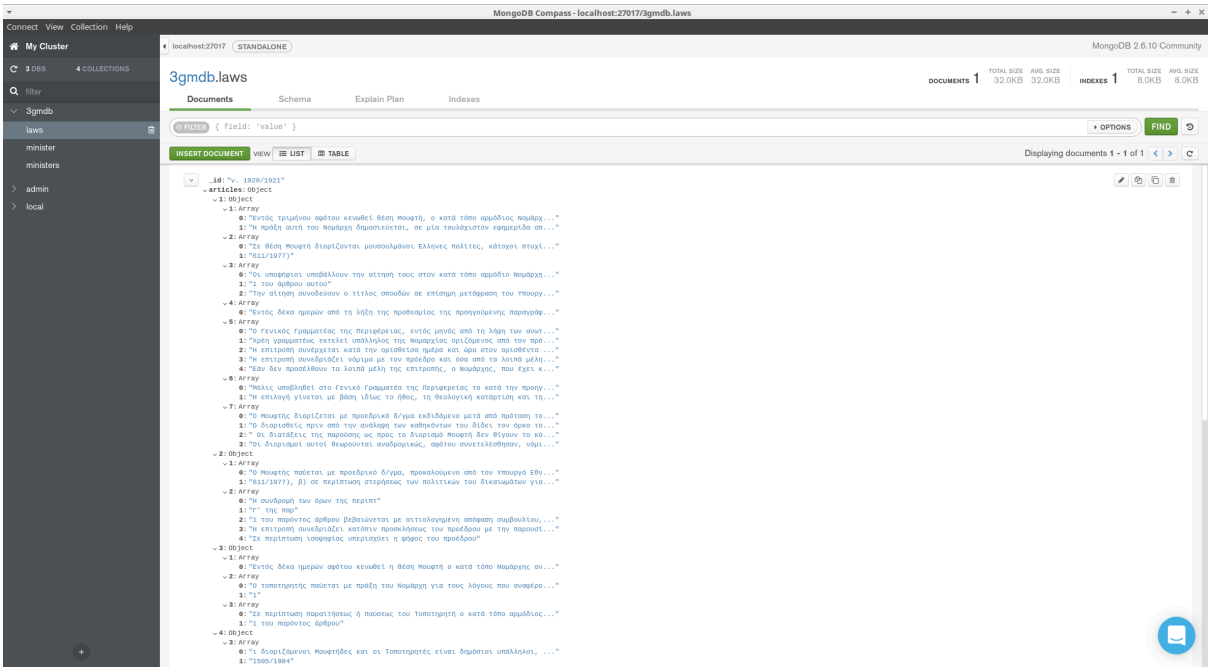
TODOs for Week 3 (14/5 - 21/5)

1. ~~Insert some laws to database as sample for codification process (e.g. Legislative Act 24.12.1990) for evaluating methods like additions~~
2. ~~Parse respective samples and insert to DB split into articles and then paragraphs~~
3. Bind relative words into sentences
4. Test results on issue 2 (15.1.2018)

Week 3 (14/5 - 21/5)

1. Bugfixes in parser.py
2. Changes in parser.py
    a. Serialization of Issues (IssueParser class) for insertion into DB. Example (17.txt / Issue 2, 15.1.2018). Example:

    b. Implementation of LawParser class to parse laws in text format from ISOKRATIS legal database and make them into a friendly serializable format (namely conversion into dict) for inserting into MongoDB organized in sentences. Example (law 1920/1921):



3. Changes in syntax.py
    a. Changes in tree hierarchy (root > law > article > paragraph > period)
    b. Nesting dictionary query (nest_tree function) so that the resulting tree has the form of

```
tree = {  '_id' : 'v. 1234/5678', 'article': { '_id' : 4, 'context' : 'Lorem Ipsum' } }
```

    c. DFS Traversal of the tree
    d. Unit testing on data/17.txt
    e. Unit testing on data/{replacer, deleter}.txt
    f. Replacements and insertions translate to collection.save() function.
4. Changes in entities.py
    a. Introduction of new entities such as Presidential Decree with regexes and made existing entities more robust by abbreviations (e.g. παράγραφος = παρ.)
5. Changes in tests.py
    a. Added more unit tests such as:
        i. test_action_tree_generator()
        ii. Test_action_tree_generator_replace_query()
        iii. test_action_tree_generator_delete_query
        iv. test_action_tree_generator_insert_and_replace():
        v. test_law_parser(filename='../data/24_12_1990_legislative_act.txt',
        vi. test_issue_serializer_to_db(filename='../data/17.txt')
6. Continuous Integration : Fix some issues with travis CI
7. Algortihm

    a. Detection of keywords (verbs)
    b. Then I search around the word in a finite window for the subject (which will also be a keyword in a nominal fall
    c. The subject gives details directly about the depth of the tree / (list). For example, if we add a paragraph, we need the law and the article.
    d. This in turn gives a dictionary with what to do and the context is run on MongoDB.
8.

TODOs for Week 4 (21/5 - 27/5)

1. ~~Move query function from database.Database to parser.LawParser to get new serializable objects.~~
2. ~~Implement deletition in the same manner as addition and replacement~~
3. Detection of phrases and periods
4. ~~Replacement / Removal and addition of phrases~~
5. ~~Unit tests on the above~~
6. ~~Fix issues with travis CI~~

Week 4 (21/5 - 27/5)

1. Code tidy-up
2. Travis CI script finally fixed
3. Amendments in README.md (include timeline and additional info)
   Added unit tests for issue parser
4. In parser.py
    a. Addition of articles / paragraphs / periods / phrases
    b. Removal of articles / paragraphs / periods / phrases
    c. Deletion of articles / paragraphs / periods / phrases
    d. Document class
5. In syntax.py
    a. Search for the exact phrase (new_phrase) in the extract
    b. Search for the old phrase in the extract using keywords
    c. Extract phrases
6. In tests.py
7. In lda.py: Implement connected components for topic modelling
8. Github Wiki initiated here https://github.com/eellak/gsoc2018-3gm/wiki

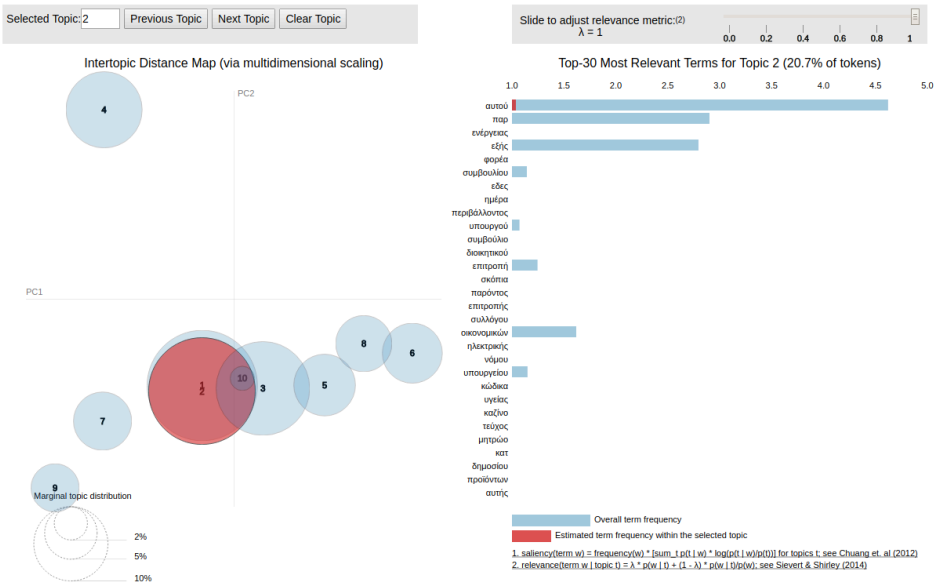
TODOs for Week 5 (28/5 - 3/6)

1. Improve detection of phrases and periods
2. ~~Implement detection of numerals in full form like (πρώτος, δεύτερος etc.)~~
3. ~~Create LawCodifier Class~~
4. More test cases
5. ~~Visualize topic modelling~~
6. ~~Document remaining examples~~
7. ~~Bug fixes~~

Week 5 (28/5 - 3/6)

1. In entities.py
    a. Convert full numerals to numbers (δέκατος πέμπτος -> 15)
    b. Inverse procedure (15 -> δέκατος πέμπτος)
    c. More regular expressions for cases lice legislative (ν.δ.) and presidential decree (π.δ.)
2. Create codifier.py for the codifier class that binds the other utilities
    a. Contains LawCodifier class that will be responsible for binding the utilities together
    b. You can test it (for the time being) via:
    python3 codifier.py ../data/testcases/17.txt
3. Integrate distutils (setup.py). Since the project changes it may occasionally not work
4. In lda.py

a. Visualize topic models with pyLDAvis



b. Improve topic modelling with Lemmatizing with lemmas from https://github.com/eellak/gsoc2018-spacy and the Hunspell dictionary (distributed under GPL license)
c. Filter out small words and everything else (such as numbers) considered “junk” for the topic modelling system.

TODOs for Week 6 (4/6 - 11/6) - Start of Exams Period

~~1. Evaluation report~~
~~2. Fetching tool~~
~~3. Code refactoring~~
4. Work more on heuristics **(will take > 1 week)**
    a. Detection of plurals
    b. Detection of subparagraphs and (subsections (?))
    c. ~~The amendment should be done in the newest statute. For example if there is an addition and the elements v. 1234/2018 and v. 1236/2007 are found we shall refer to the newest statute, i.e. v. 1234/2018~~
    d. Bug Fixes on existing methods

Week 6 (4/6 - 11/6) (Exam Period)

1. Fetching tool
2. Detect amendments per statute in order to codify per law and not per issue (see parser.py)
3. Change DB structure to hold each amendment as a record under the same key.
    a. In database.py:
        i. Append each serializable under versions
        ii. Detect based on version number
        iii. Avoid the same amendment being done for the second time
    b. In codifier.py:

    i. Populate Laws on their last version
    ii. codify_law(identifier)
4. Code refactoring
5. New datasets to be added (all Government Gazette Issues from ET).
6. Wiki enrichment - Moved readme
7. Installation instructions

TODOs for Week 7 (12/6 - 17/6) (Exam Period)

1. ~~Finish restructuring the db~~
2. ~~Pretty print - Stringify of laws. TeXifier~~
3. Insert (some) FEKs in DB
4. Repo Cleanup
5. ~~Detect new laws in GG articles. New laws can be detected as follows in a GG article~~

ΝΟΜΟΣ ΥΠ' ΑΡΙΘΜ. 1234

Εκδίδομε τον ακόλουθο Νόμο που Ψήφισε η βουλή

Άρθρο 1

Lorem Ipsum

Week 7 (12/6 - 17/6) (Exam Period)

1. Database restructuring (each version of the law is kept on the database)
2. In parser.py
    a. Detection of new laws inside FEKs (tested on 2018 and 2017 FEK A)
    Example:
    Εκδίδομε τον ακόλουθο Νόμο που Ψήφισε η βουλή
    Άρθρο 1
    Lorem Ipsum
    b. Fix paragraphization (?) issues
3. In codifier.py
    a. Changes in the API of LawParser
    b. TeXification tool with xelatex for pretty output
4. In tests.py
    a. Removal of legacy tests
5. Scripts
    a. Batch conversion from pdf to txt with pdf2txt.py from pdfminer under scripts/converter.py
    b. OCR converter that uses the tesseract ocr engine https://github.com/tesseract-ocr/tesseract with Greek, English and Classical Greek pre-trained models. Script under scripts/ocr.py. This script is used to convert legacy GG documents that are in image format.
6. Miscellaneous

a. Datasets from GG documents (FEK A) from 1974 to 2018 from the ET
b. Migration to Git LFS for large files
c. Cron job for fetching new documents from ET every day using scripts/fetcher.py
d. Conversion of GG articles to raw text from 2000 to 2018. Dataset under data/dataset.zip

TODOs for Week 8 (18/6 - 25/6)

1. ~~Run OCR on all GG documents on stereo vm. Resolve bugs such as memory errors etc.~~
2. ~~Document scripts/*~~
3. ~~Document end 2 end proceedure~~
4. ~~Finish Presentation for FLOSS meetup~~
5. ~~Document LawCodifier class~~
6. ~~Markdown Support for exporting (in order to export html, latex etc.)~~
7. ~~Argparse support for scripts/~~
8. **Change generate_action_tree() structure**
    a. Per paragraph in law
    b. Get string as argument instead of issue, etc.
    c. Generate extracts and non-extracts inside function
    d. Check with examples all possible combinations

Week 8 (18/6 - 25/6)

1. Run OCR on all GG documents on stereo vm.
2. Resolve bugs such as memory errors etc. in scripts/ocr.py that were due to image magick
3. Document scripts/*
    a. Write documentation for converter.py, fetcher and ocr.py in the wiki
    b. https://github.com/eellak/gsoc2018-3gm/wiki/Fetching-Documents
    c. https://github.com/eellak/gsoc2018-3gm/wiki/Document-Processing
4. Document end 2 end proceedure here: https://github.com/eellak/gsoc2018-3gm/wiki/Tutorial
5. Finish Presentation for FLOSS meetup
6. Document LawCodifier class
7. Markdown Support for exporting (in order to export html, latex etc.)
8. Argparse support for scripts/{converter, ocr, fetcher}.py (also described in documentation)
9. **Change generate_action_tree() structure**
    a. Per paragraph in law
    b. Get string as argument instead of issue, etc.
    c. Generate extracts and non-extracts inside function
    d. Check with examples all possible combinations

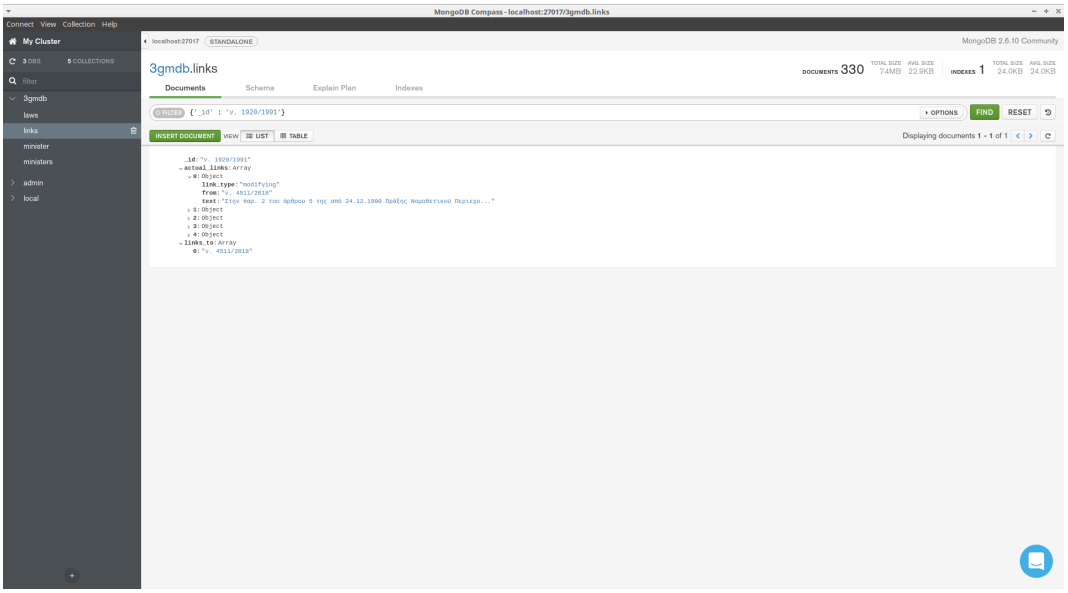
Week 9 (26/6 - 1/7)

1. Sync data with Pithos+
2. Demo of project Syntactic Analysis to test amendment detection as Flask Application under demo/app.py. Used for testing purposes as well.
3. Changed bracketization method in helpers.py
4. Wrote custom tokenizer at 3gm/tokenizer.py for splitting avoiding abbreviations such as παρ. ν., π.δ. etc.
5. Cross-linking visualization with NetworkX under 3gm/grapher.py
6. Modification of scripts/fetcher.py to work in headless environments.
7. Use spaCy to perform NLP on sentences and detect subjects (far better than previous approach with window).
8. Bugfixes in syntax.py

Week 10 (2/7- 9/7)

1. Visualization script for graph generation at grapher.py
2. Export docs with pydoc3 under docs/
3. Repository cleanup
4. In demo app:
    a. Add codification page
    b. Add links between laws
    c. Add autocomplete in searchbar
    d. Add bootstrap support
    e. Render law from md in page
5. In helpers.py
    a. Added helper scirpts:
        i. SSCONJ iterators
        ii. Whitespaces / hyphenthation fixes
6. In codifier.py

a. Add Link class to display links between laws (either modifying or referential) with support for database



TODOs for Week 11

1. ~~Populate db with more laws~~
2. Status in links => apply links
3. ~~Display all versions in demo app (Separate page / Same page ?)~~
4. ~~Transform plurals~~
5. ~~Improve phrase detection~~
6. ~~Hyphen fix!~~
7. ~~Code cleanup and refactoring~~

Week 11 (9/7 - 15/7)

1. Filter out subordinate conjunctions in tokenizer.py
2. Greek translation of web application / demo
3. Add rollback and checkout support in database.py
4. Split syntax trees functionality in syntax.py
5. Greek numerals support in entities.py
6. Template filters for jinja2 (markdown rendering, badge rendering etc.) in app.py
7. Add full linking index with ordered links between statutes
8. Refactor code in topic_models.py
9. Add doc2vec training scripts for similarity analyzer with doc2vec on statuses corpora
10. Implement history page https://github.com/eellak/gsoc2018-3gm/issues/3
11. Add issue-like format in exporting tools
12. Document modules (parser, codifier)
13. Pep8 refactoring using autopep8
14. Change CLI interface in codifier CLI tool
15. Improve CLI tools
    a. Add multiprocessing support
    b. Add more options
    c. Transform needed tools to UNIX philosophy https://github.com/eellak/gsoc2018-3gm/issues/6
    d. Add bare exporter tool
16. Improve acurracy of syntax tool

Week 12 (16/7 - 22/7)

1. Improve syntax tool
    a. Rewrite functionality such as content fetching, level building etc. The process is outlined here: https://github.com/eellak/gsoc2018-3gm/wiki/Algorithms-for-analyzing-Government-Gazette-Documents
    b. Add support for multiple amendments
2. Separate and move tools to 3gm/tools
3. UI Enhanchements in web app
    a. Add effects in homepage

    b. Improved CSS
4. Add separate phrase module for phrase processing.
    a. Defined regular languages for detection of phrasal amendments. Module located at phrase_fun.py Examples:
        i. subj_before_phr = r'πριν (από |)(τη φράση|τη λέξη|τις λέξεις)[^«]«[^»]*»'
        ii. subj_after_phr = r'μετά (από |)(τη φράση|τη λέξη|τις λέξεις)[^«]«[^»]*»'
        iii. subj_rep_phr = r'(με |από)(τη φράση|τη λέξη|τις λέξεις)[^«]«[^»]*»'
        iv. subj_phr_regex = r' η (ακόλουθη φράση:|ακόλουθη λέξη:|φράση|λέξη)[^«]«[^»]*»'
    b. Documented module
5. Add working examples in examples/examples.md
6. Replace old tests with new tests in tests.py
7. Migrate big files to lfs
8. Add GridFS support for storing large records (database.py)
9. Help file in web app according to https://github.com/eellak/gsoc2018-3gm/issues/7
10. Add ranking system in laws using the PageRank algorithm provided by networkX.
11. Add minimal RESTful API in web application using flask-restful. Supported endpoints are located here: https://github.com/eellak/gsoc2018-3gm/wiki/RESTful-API
12. Deploy web application to https://3gm.ellak.gr using nginx + gunicorn + supervisor.
13. Refined + Create Wiki Pages: https://github.com/eellak/gsoc2018-3gm/wiki
14. Started https://archive.org/details/greekgovernmentgazette with mentor D. Spinellis to host government documents on the internet archive.

Week 13 (23/7 - 29/7)

1. Add case and subcase support (all operations in arbitrary depth) w. integration
2. Add more tests to tests.py
3. Code refactoring
4. Separate API Endpoint for history
5. Bug fixes
6. Search supports topics by rank (provided via PageRank) or chronological ordering
7. Improve versioning adding checkout support
8. Improve wiki documentation
9. Integrate IA (https://archive.org/details/greekgovernmentgazette) project with webapp

Week 14 (30/7 - 5/8)

1. Bind final project and upload to https://3gm.ellak.gr
2. Fix deployment issues (SSL certs etc.) on host vm
3. Improve wiki documentation
4. Finish apply_links.py to generate law versions
5. Create summarizer module for generating summaries from titles located at 3gm/summarize.py
6. Finish writing docstrings

7. Web app enchangements
8. Bug fixes
9. Diff tool for laws in webapp with difflib. Example: https://3gm.ellak.gr/diff?initial=%CE%BD.+4009%2F2011&identifier=%CE%BD.+4009%2F2011&final=%CE%BD.+4485%2F2017

Week 15 (6/8 - 12/8)

1. Last minute bug fixes
2. Amendment removal algorithm