Efficient Management and Optimization of Very Large Machine Learning Dataset for Question Answering

Marek Medveď Radoslav Sabol Aleš Horák
xmedved1@fi.muni.cz, xsabol@fi.muni.cz, hales@fi.muni.cz
NLP Centre, Faculty of Informatics, Masaryk University

December 7, 2020

ZODB

- transparent Python-object persistence (database) system
- stores selected data models (especially classes and objects in the Python)
- store huge hierarchical structures that are not limited to one data type

SQAD to ZODB

- SQAD database transformed into ZODB system (main storage of all the dataset data)
- ZODB allows fast access to the SQAD data and efficiently stores all data from the SQAD database raw records in the final form required by the AQA question answering system
- allows direct access to data – differs from standard approach
- stores Python objects without a need of extra format conversion
- stores the complete SQAD database records

SQAD record No. 012878

Original text: *Ngoni (někdy též n'goni) je strunný hudební nástroj oblíbený v západní Africe. ...*
[Ngoni (also called n'goni) is a string musical instrument popular in west Africa.]
Question: *Jakého typu je hudební nástroj ngoni?*
[What kind of musical instrument is ngoni?]
Answer: *strunný hudební nástroj*
[string musical instrument]
URL: `https://cs.wikipedia.org/wiki/Ngoni`
Author: *login*
Question type:*ADJ_PHRASE*
Answer type: *OTHER*
Answer selection: *Ngoni (někdy též n'goni) je strunný hudební nástroj oblíbený v západní Africe.*
Answer extraction: *strunný hudební nástroj*

Additional features

- **word vectors** – to boost the training procedure in the AQA answer selection module the new SQAD-ZODB database stores pre-computed word vectors pre-trained from large Czech corpora using the word2vec algorithm.
- **list of sentences containing exact answer** – during building SQAD-ZODB, the list of sentences that contain the exact answer is computed. This information is then used in the evaluation process of the answer selection module.
- **list of similar answers** – boost the module ability to identify the correct answer within a list of very similar sentences.
- **answer context** – to supplement the neural network decision process,an information about the sentence context is provided to the answer selection module. The SQAD-ZODB database contains several types of context pre-computed from the original data.

SQAD-ZODB context

- previous sentences – the context of N full sentences is added to each input article sentence.
- phrases from previous sentences – using the rule-based SET parser, the system is able to identify all possible noun phrases within each sentence. M noun phrases from each of N preceding sentences are stored as the second context type.
- “link named entities” from previous sentences form the third type of sentence context.

Link named entities

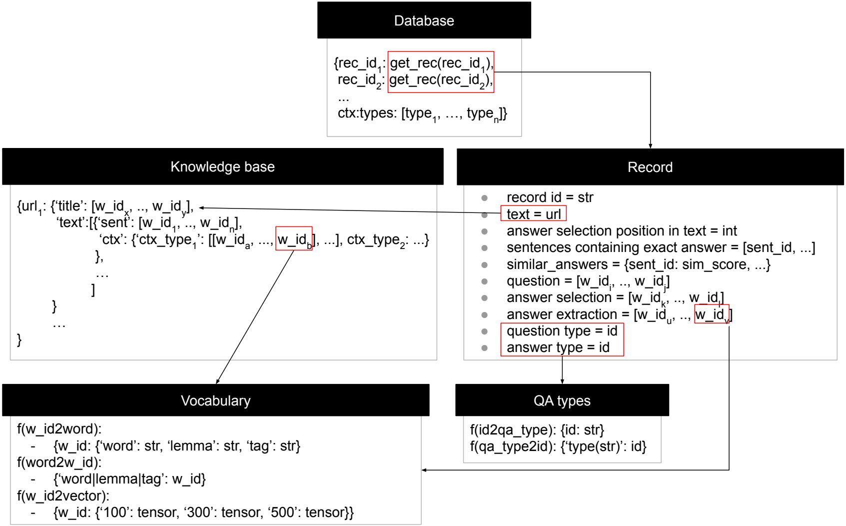
- LNEs are defined as entities that are labeled with Wikipedia internal links
- inside each Wikipedia article, links that refer to other Wikipedia articles identify entities which are often significant in denoting an important piece of information
- named-entity recognition (NER) system – `https://github.com/kamalkraj/BERT-NER`
- final module is applied on SQAD data and provides information about recognized link named entities which are used as a sentence context

Link named entities – training data

word	NE tag
...	
přestoupil	O
do	O
Sparty	B
Praha	I
...	

- O – regular word
- B – beginning of named entity,
- I – continuation of named entity.

SQAD-ZODB architecture



SQAD-ZODB architecture (DATABASE object)

- first level, the SQAD-ZODB database stores all records IDs and a function that builds the record content form 4 database parts (tables)
- **Record**

SQAD-ZODB architecture (Record object)

- Record ID - unique identifier of a SQAD record
- Text variable - contains a unique *URL* that points to specific article inside the *Knowledge base* table (de-duplication)
- Answer selection position – stores an index of the sentence that contains the expected answer
- Sentences containing exact answer – is a list of sentence IDs that contain the exact answer
- Similar answers – list of similar sentences with their similarity scores
- Question, answer selection and answer extraction – lists of words IDs. Each word ID can be transformed into word, lemma, POS tag, 100-, 300- or 500-dimensional vector using the *Vocabulary table*.
- question type and answer type – IDs pointing to specific question and answer type using the *QA types table*.

SQAD-ZODB architecture (Knowledge base object)

- stores all articles used within the SQAD database
- avoid duplicates
- stores only list of the words IDs – compact size while maintaining all important information

Updating the SQAD-ZODB Database

- transaction support in ZODB
- each new feature is designed as standalone script that can supplement the database with a single new feature of each record
- list of available transformation scripts:
  - sqad2zodb
  - add_similar_sentences
  - add_sentences_containing_exact_answer
  - context_previous_sentences
  - context_noun_phrases
  - context_ner

SQAD-ZODB Performance

- direct access to required information - saves time and amount of transfered data

Representation	Disk usage
Plain text	1,312.89 GB
Pickle	240.20 GB
SQAD-ZODB	25.08 GB

SQAD-ZODB Performance

Preloaded vocabulary	
init	12.44
w	13.21
l	2.02
t	1.42
v1	4.78
v3	2.61
v5	2.61
w;l;t;v1;v3;v5	4.03

SQAD-ZODB over Network

- SQAD-ZODB database was implemented within the ZEO[1] (Zope Enterprise Objects) library that allows to run the database in the client-server mode over network
- hyperparameter optimization of large amount of training setups

[1] `www.zodb.org/en/latest/articles/old-guide/zeo.html`

Optuna

- recent hyperparameter optimization framework
- highly scalable
- supports multiple database engines (SQLite, PostreSQL, MySQL)
- **define-by-run** API allows adjustments to hyperparameter ranges across runs
- built-in support for **pruning**

Comparison of HP Optimization Frameworks

2

Framework	API Style	Pruning	Lightweight	Distributed	Dashboard	OSS
SMAC [3]	define-and-run	✗	✓	✗	✗	✓
GPyOpt	define-and-run	✗	✓	✗	✗	✓
Spearmint [2]	define-and-run	✗	✓	✓	✗	✓
Hyperopt [1]	define-and-run	✗	✓	✓	✗	✓
Autotune [4]	define-and-run	✓	✗	✓	✓	✗
Vizier [5]	define-and-run	✓	✗	✓	✓	✗
Katib	define-and-run	✓	✗	✓	✓	✓
Tune [7]	define-and-run	✓	✗	✓	✓	✓
Optuna (this work)	define-by-run	✓	✓	✓	✓	✓

Figure: Comparison of hyperparameter optimization frameworks in terms of available features.

[2] Source: https://arxiv.org/pdf/1907.10902.pdf

Optuna and AQA Answer Selection

- AQA Answer Selection's hyperparameter search space is getting increasingly more complicated as more new features/parameters are added to the module
- hyperparameter optimization methods used for AQA before Optuna:
  - Grid Search: performance requirements increase exponentially with each added hyperparameter
  - Manual Search: works well, but cannot be automated easily :-)
- for abovementioned reasons Optuna was integrated into Answer Selection as an automated method of optimizing the hyperparameters

Search Space Definition

Table: Hyperparameter values search space for the answer selection model

Hyperparameter name	Optuna distribution used	Range of values
BiGRU hidden size	discrete uniform	100-600 with the step of 20
Dropout	discrete uniform	0.0-0.6 with the step of 0.1
Batch size	categorical	1, 2, 4, 8, 16, 32, 64, 128, 256
Optimizer	categorical	Adam, Adagrad, Adadelta, SGD
Learning rate	logarithmic uniform	from 10^{-4} to 10^{-1}
Embedding dimension	categorical	300, 500

Setup Details

- **PostresSQL** database used as storage for trial results
- all trials were computed remotely on **Metacentrum** machines
    - training data was transferred either locally with **ZODB** or over network with **ZEO**
- pruning of unpromising trials with **MedianPruner**
    - prune trial if it's validation accuracy at epoch *e* is worse than median of validation accuracies of previous trials at epoch *e*
- Tree Parsen Estimators (TPE) used as optimization mechanism

Results

- 1506 trials were trained
  - 455 were succesful, 365 were pruned, and the rest has crashed due to GPU OOM errors
- the best run reaches the **MAP** of 83.13 %
  - an increase of 0.8 %
- the new best hyperparameter setup
  - **Embedding Dimension:** 300
  - **Dropout Probability:** 0.3
  - **Optimizer:** Adagrad
  - **Batch Size:** 4
  - **Learning Rate**: 0.0042
  - **BiGRU Hidden Size:** 520

Results

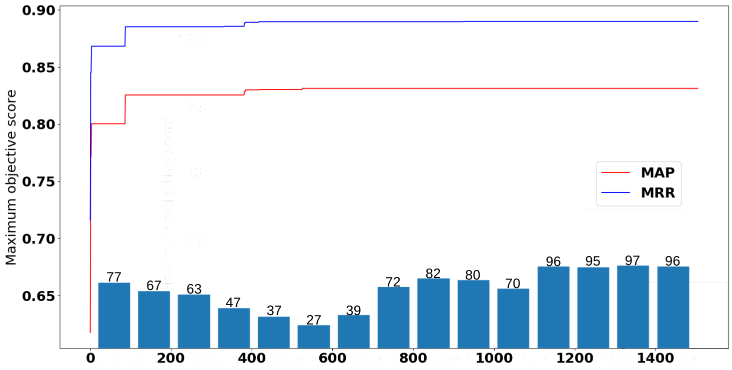


Figure: Incrase in MAP and MRR during all 1,506 recorded runs. Histogram below depicts the number of pruned runs

Conclusion and Future Work

- managing the efficient storage and data transfer of very large QA dataset
- Optuna improves model MAP of about 1%

Future work:

- test new types of context
- decrease the amount of erroneous trials in optimization – eliminate the bias towards lower embedding dimensions

Thank You for Your Attention!

MUNI
FACULTY
OF INFORMATICS