Proceedings of the 1st International Workshop on Language Technology Platforms (IWLTP 2020), pages 54–58
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

The Kairntech Sherpa – An ML Platform and API for the Enrichment of (not only) Scientific Content

Stefan Geißler
Kairntech SAS
29 Chemin du vieux Chêne, 38240 F-Meylan, France
stefan.geissler@kairntech.com

Abstract

We present the Sherpa Platform and API that combines various ML and NLP approaches for the analysis and enrichment of textual content. The platform's design and implementation is guided by the goal to allow non-technical users to conduct their own experiments and training runs on their respective data, allowing to test, tune and deploy analysis models for production. Dedicated specific packages for subtasks such as document structure processing, document categorization, annotation with existing thesauri, disambiguation and linking, annotation with newly created entity recognizers and summarization – available as open source components in isolation – are combined into an end-user-facing, collaborative, scalable platform to support large-scale industrial document analysis. We see the Sherpa's setup as an answer to the observation that ML has reached a level of maturity that allows to attain useful results in many analysis scenarios today, but that in-depth technical competencies in the required fields of NLP and AI is often scarce; a setup that focusses on non-technical domain-expert end-users can help to bring required analysis functionalities closer to the day-to-day reality in business contexts.

Keywords: Natural Language Processing, Machine Learning, End-user software

1. Introduction

Machine Learning (ML) approaches have been able to go beyond the previous state of the art results in many different fields in the recent years and tasks in natural language processing (NLP) are no exception here. In scenarios such as machine translation, speech recognition, entity recognition, sentiment analysis, document categorization and many others, ML has proven to deliver the highest quality in many evaluations (Chollet, 2017).

At the same time ML comes with its own set of requirements such as the need for technical expertise in programming and data science as well as the necessity to prepare appropriate volumes of training data. Both these requirements can put a heavy burden on the application of ML in business contexts where data science expertise is scarce and costly and training data often not available in the right quality and formats. As Neven and Seva (2019) emphasize: "Manual annotation is still regarded as the bottleneck for many NLP experiments, given that it is a time-consuming manual process."

We present the Kairntech Sherpa, a web-based collaborative platform for ML that allows to address many NLP requirements and that at the same time can be operated by domain experts and end users with little or no technical data science expertise. Users can train, evaluate, tune and deploy ML models for subsequent use via an API in industrial document analysis scenarios.

2. Document Analysis Subtasks

NLP subtasks such as document structure recognition, entity recognition, document categorization, thesaurus-based indexing or summarization have not only been areas of active research for many years but they also have a firm place in business needs around the management, the digestion and distribution of text-based content in large industry organizations.

The Sherpa gives the user access to these functionalities; we go through each of these in the subsections below.

2.1 Document Categorization

Assigning a document to one or several of a predefined set of categories is a task that has its place in a wide range of document analysis scenarios; it also is a well-studied topic in the NLP field. The Sherpa offers users to either upload a precategorized corpus into the application or to upload uncategorized content and then add the categories manually and then to train a model. There is a broad range of categorization algorithms available in the public domain and while the display, the training and evaluation of document categorization is an important feature of the Sherpa API, the precise choice of the underlying algorithm may vary – at the time of the writing of this document categorization via the Python scikit-learn library[1] as well as approaches based on a deep learning library[2] are offered.

2.2 Thesaurus-based Indexing

Annotating ("indexing") document with a set of appropriate descriptors from a set of hierarchically structured terms is another well-established technique in information management, where automatic approaches have been studied and applied for many years and with great success. (Medelyan and Witten, 2006)

Automatic indexing needs to cope with a range of requirements beyond merely finding the occurrence of a string in the text: terms often occur with variations due to inflection, terms may be ambiguous (the same string can carry different meanings depending on the context), terms vary with respect to their importance from terms with only a peripheral role in the document to those that represent the core topic of a document. Finally, where terms are associated with background information, automatic indexing benefits from linking the occurrence of the term to this background information, thus enriching the document with information that is not originally part of the text but that is often of additional relevance to the reader. A typical example is the place (geo coordinates) of a location displayed on a map or background information for

[1] https://scikit-learn.org/stable/

[2] https://github.com/kermitt2/delft

a company (website, logo, etc) or a person (picture, address).

The Sherpa employs the "entity fishing" library[3] that uses more than 78 mio terms from Wikidata[4] to enrich content. Wikidata is a superset of many widely used domain-specific thesauri – the well-known MeSH[5] that is used for indexing medical content for instance, is a subset of Wikidata. While the approach can work in principle with Wikidata knowledge bases in any language, we have chosen to add by default the resources for only a selection of languages[6]. There is no technical reason for that decision, it is rather a matter of striking a balance between effort and disk space on one side and the demand from Sherpa use cases on the other.

Wikidata is constantly evolving and growing and we have put processes in place that allow the indexing approach inside Sherpa to keep up to date with this growth. While this is not yet fully automated (partly due to the considerable size and compilation requirements that are needed to turn Wikidata into the format deployed as part of the Sherpa), the application nevertheless benefits from regular updates prepared by the Kairntech development team.

2.3 Custom ML Annotators

Even with many pre-trained models and existing thesauri and term lists in the public domain or available within a given organization, often a given new task just requires setting up a new annotator from scratch in order to properly address a new requirement. Training a new annotator however can be costly: manipulating corpora and setting up and tuning sophisticated ML algorithms is a task that requires a certain level of precious data science expertise that may be scarce and even if that expertise is available, preparing a proper corpus often means conducting time-consuming corpus annotation efforts, which sometimes mean efforts of many days or more.

A prime focus in the design of the Sherpa was therefore making this process of annotating content as easy and effortless as possible.

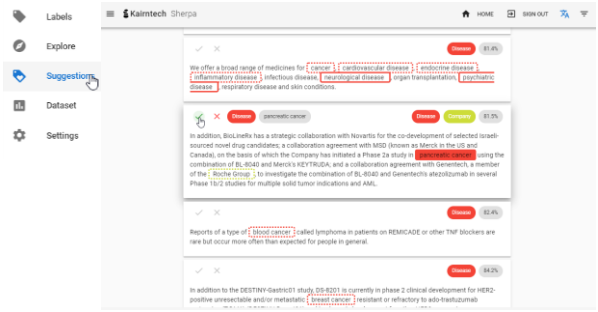

Figure 1: The Sherpa GUI presents the content as easy-to-consume snippets of text to be annotated by the user. Asking for new "suggestions" applies the current ML model which is continuously refined in the background.

Various implementation details support the manual annotators in proceeding with their tasks as quickly as possible: For instance the boundaries of the to-be-selected expressions are automatically extended to include the leftmost and rightmost word boundaries of the selection, respectively which frees the user from the burden of having to accurately hit these boundaries with the mouse herself. Also, after a given snippet is properly annotated (or when the user asks for new "suggestions") this new list is not presented in a random order (or just alphabetically) but instead an Active Learning scheme is applied (Settles, 2009) that ensures that the system focusses in particular on those examples that promise the highest learning progress. It has been observed that properly implemented Active Learning schemes can reduce the effort for manual

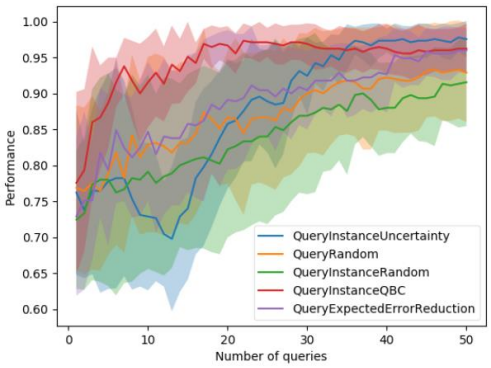

Figure 2: Chart showing how quickly training success on a categorization task improves in accuracy under different training example selection schemes.

annotation by up to 93% (Laws, 2013).
In the chart above we see different training sample selection schemes improving their performance on a categorization task, the well known "iris dataset"[7]: The naïve, random selection scheme (green line) rises comparably slow while one Active Learning approach ("QBC" – Query by committee, the red line) arrives at high accuracy level much more quickly. Translated into project efforts, this can mean drastic reduction of manual annotation efforts.

2.4 Document Structure Recognition

Many types of documents that are relevant in a business context today have a somewhat formal, fixed structure: Contracts, scientific papers, tech reports, invoices or patents typically have a fixed set of chapters and a type-specific way to present certain key metadata to the reader that is meant to facilitate reading and the digestion of the content. However, this information is often lost when the document is rendered into unstructured formats like the notorious PDF. For instance, in the process of writing a LaTeX[8] document, the information what a document's author or title is or what the names of the cited authors are is explicitly marked up; however this information is most of the time no longer present explicitly in the final PDF. The human reader can easily parse this PDF making use of visual clues like fonts and formatting but in order to be

[3] https://github.com/kermitt2/entity-fishing
[4] https://www.wikidata.org/
[5] https://www.nlm.nih.gov/mesh/meshhome.html
[6] German, English, French, Italian, Spanish and Dutch
[7] https://archive.ics.uci.edu/ml/datasets/Iris
[8] https://www.latex-project.org/

made available for subsequent document management processes this information must be recognized and extracted.

We use the Grobid[9] package to automate the processing and recognition of unstructured documents to reconstruct their structure and meta data for these tasks. The result of the processing of a document with Grobid is a TEI XML[10] document that makes information about the document's title, authors, their affiliations, the chapter structure, date, references and many others explicit.

Recognizing the structure of a document is again one of these tasks that appear easy for the human reader but that turn out to be hard to capture into explicit rules. Grobid therefore is based on a document-type-specific training corpus capturing text-based but also layout-based information and relying on an appropriate training corpus. The code and the accompanying models referenced above are set up to handle scientific documents; in order to handle another document type like, say, contracts, a new Grobid model would need to be generated. This adaptation of the Grobid component is currently not yet supported via the Sherpa GUI but must be carried out externally.

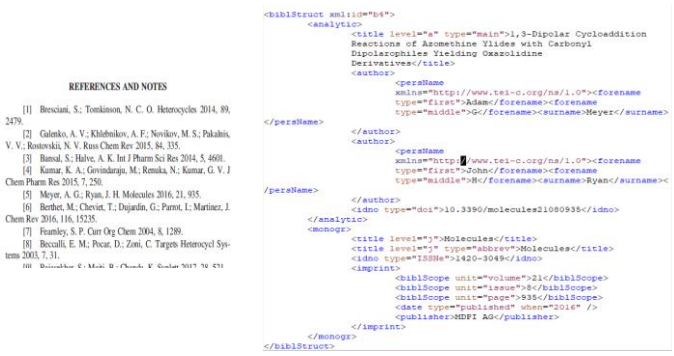

Figure 3: Example for document structure recognition.

The example in Figure 3 shows a part of the reference section of a scientific paper, first in the unstructured PDF, then as structured XML (TEI) after the processing by Grobid: each cited paper and each author is wrapped into the appropriate XML element. The author is optionally dereferenced, disambiguated and completed through a lookup in resources such as CrossRef[11].

Note, that while the Sherpa is a comparably recent development, Grobid in isolation has in fact already been deployed in production in quite a number of large scale installations such as ResearchGate, the European Patent Office EPO, the CERN, the INIST and others.

3. Tracking Quality

For some of subtasks listed above, the Sherpa offers different choices with respect to the used algorithm: For entity recognition / sequence labelling for instance users can decide between different options such as an implementation of Conditional Random Fields (CRF) or libraries implementing deep learning approaches (e.g. Spacy[12], Delft[13] or Flair[14]). Different approaches may differ significantly in their behavior and appropriateness for a given task: A CRF is trained comparably fast while the results are often a few percent or more behind those of slower Deep Learning runs.

The Sherpa provides users with an overview of the development and the latest training successes of the various employed options. Users can also rely on the fast CRF approach to quickly refine a model that constantly presents new text snippets for manual annotation and once enough snippets have been annotated, launch a longer training run with the more resource-intensive Deep Learning libraries. This way of combining the various strengths and weaknesses of different ML approaches would normally require considerable technical ML expertise – we have chosen to offer that to also less or non-technical users as part of the Sherpa user interface.

Figure 4: The Sherpa GUI provides the user with an overview about the respective quality reached by different algorithms launched on the same task.

Besides illustrating the fact that the Sherpa allows users to run and compare multiple experiments on the data of a project, the picture above also illustrates the quality delivered by the underlying Delft machine learning library. While a detailed evaluation of Delft is beyond the scope of this paper, the respective results and comparisons to other approaches can be studied at the Delft project page[2].

4. The Sherpa REST API

All the interactions of the user on the GUI are backed by a respective REST API call. That means that while the GUI is the preferred way to conduct an annotation campaign in the browser, the Sherpa can easily be integrated into third party environments via the API, either with the complete training, management and prediction use cases or for instance with only the prediction part.

The Sherpa API is available for inspection at https://sherpa.kairntech.com/swagger-ui/. Note that authentication is required in order to actually use it. While we are currently not in the position to make the API or the Sherpa (https://sherpa.kairntech.com/sherpa/signin) freely

[9] https://github.com/kermitt2/grobid
[10] https://tei-c.org/
[11] https://www.crossref.org/
[12] https://spacy.io/
[13] https://github.com/kermitt2/delft
[14] https://github.com/flairNLP/flair

accessible, we are always open to provide access for testing and evaluation on request[15].

5. Collecting User Feedback

Since the Sherpa is in particular targeted towards end-users, collecting end-user feedback was an important item on our agenda early on.

We had invited professionals on two occasions to a "Hackathon" in July 2019 (20 users) and Oct 2019 (50 users), respectively, introduced them to the concepts and ideas behind the Sherpa and encouraged them to execute their own training experiments.

The key feedback we collected was that users support the claim by the Sherpa, that the annotation of text corpora – normally not the most cherished task of information professionals and related experts – is facilitated and supported favorably by the application. The way in which the GUI helps to minimize mouse movements and keystrokes when doing larger amounts of annotations was considered an important time-saver by users, many of which had previous experiences with training corpus preparation. The most important aspect for many however was the way in which users get constant, live feedback by the system as they go along. The fact that the Sherpa continuously uses the information created so far to refine and apply the model and was considered useful and motivating. One user told us "it even makes you want to annotate more" while another used the term "addictive".

The evidence above, however, remains anecdotal and needs to be complemented with more quantitative data and larger numbers of users as the adoption of the platform grows.

6. Related Work

The booming popularity and continued success of ML-powered NLP led also to an increase in available NLP platforms that claim to wrap the complexities of ML underneath an end-user GUI. A comprehensive overview of the available systems is hard to achieve given the fast pace of developments in this field. Neves and Seva (2019) have presented such an overview together with an evaluation of the identified systems. They apply a set of criteria that may not be appropriate in industrial context: One of their criteria is that the studied system be available to them, another one that installation must be possible, again for them, in under 2h. These criteria can be defended in order to keep the effort in conducting a scientific study manageable, but they evidently limit the range of studied systems and are not entirely relevant in an industry context[16].

In their list, the authors identify several systems that follow a similar direction as the Sherpa, namely to combine easy corpus annotation directly with ML capabilities using the annotations to create and refine underlying ML models. Example here are Prodi.gy[17], tagtog[18] or LightTag[19].

Seen the list of existing text annotation environments above, the motivation for adding with the Kairntech Sherpa yet another one requires some explanation. Some tools like tagtog or webanno allow richer annotations like e.g. adding metadata on entities at the cost of making the application more complex for the kind of use case we had in mind. The guiding principles for the Sherpa were first of all speed of annotating content and the minimization of the mouse movements and buttons to press when stepping through corpus. Also, the direct integration with an underlying model that constantly learns as the user proceeds for user interaction was key. At any moment the user can request a new result to verify and curate, based always on the most recent model.

Verification is often much faster than adding annotations from scratch. Users find themselves quickly jumping in quick succession between adding annotations and applying the latest model to yet unannotated text. This not only speeds up annotation but moreover is perceived as rewarding by the users who see the automatically created results getting better as they proceed.

The perhaps broadest overlap of an existing tool with the process we felt we needed can be seen in the case of Spacy and its annotation extension Prodi.gy. While Spacy/Prodi.gy are exceptionally well designed pieces of software, some of the scenarios there rely on scripting in python. This however, while evidently greatly extending their reach, can be expected to intimidate the kind of users we have in mind for the Sherpa, i.e. domain experts with no experience or desire to dive into Python programming.

With an annotation process like this in mind and after inspecting existing tools, we concluded that none of them offered a workflow as the one we had in mind.

7. On Commercial Software and Open Source

Several of the tools listed in the study above are available, at least partially, as open source systems. Not only tools coming from a predominantly academic background but also tools implemented by commercial players often come in limited, feature-reduced versions as open source, offering license-based options for larger, industrial installations.

The Kairntech Sherpa is also a commercial tool. Key components inside, however, are available without any restriction as open source, several of them implemented by members from the Kairntech development team, e.g. Delft, Grobid and Entity Fishing[20].

8. Sample Sherpa Deployments

The range of functionalities listed above suggests that the Sherpa platform may address requirements from different industries and on different topics. We briefly describe two

[15] Enquiries can be addressed at info@kairntech.com

[16] For instance "availability" for a commercial client evidently does not mean the tool must be available free of charge on the internet.

[17] https://prodi.gy/

[18] http://www.tagtog.net/

[19] https://www.lighttag.io/

[20] Disclaimer : The key implementer behind the open source systems Delft, EntityFishing and Grobid is part of the Kairntech software development team.

use cases where the Sherpa has been selected by industrial users:

Inside the German Pharma company Boehringer Ingelheim, a dedicated group, the *Scientific Information Center*, is charged with the analysis and diffusion of scientific and market information to internal users. Boehringer has decided to deploy the Sherpa to support these processes[21]. Another scenario, relying largely on the Sherpa capacity for named entity recognition is addressed at Sealk.co whose mission is to scan large volumes of business news for information that is relevant for the topic of Mergers&Acquisitions.

9. Future Work

The Sherpa is ongoing development project and we plan to extend it continuously to cover more and more functionalities. Extending the analysis to the processing of relationships and integrating analysis results with Graph Databases is high on the agenda. A planned step for later in 2020 is the integration of the Sherpa into the ELG platform[22] allowing users to build their own analysis models on ELG content.

10. Conclusion

We have presented the Sherpa – a platform for the creation of ML training corpora, the training, evaluation and optimization as well as the deployment of the resulting models via a REST API. Technical subtleties of the use of ML approaches are "hidden" as much as possible underneath a simple user interface to allow non-technical users and domain experts to proceed using the system without the need for detailed ML background or any coding at all.

11. References

Chollet, F. (2017): Deep Learning with Python. Manning Publications.

Laws, Florian (2013) : "Effective active learning for complex natural language processing tasks.". Dissertation Univ of Stuttgart, http://dx.doi.org/10.18419/opus-3009

Medelyan, Olena, and Ian H. Witten (2006) : "Thesaurus based automatic keyphrase indexing." *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*.

Neves, M. and Seva, J (2019): An extensive review of tools for manual annotation of documents. Briefings in Bioinformatics, https://doi.org/10.1093/bib/bbz130

Settles, Burr (2009) : *Active learning literature survey*. University of Wisconsin-Madison Department of Computer Sciences.

[21] https://www.kairntech.com/articles/dec2019.html

[22] https://www.european-language-grid.eu/