Abstract

We present small-text, a simple modular active learning library, which offers pool-based active learning for text classification in Python. It comes with various pre-implemented state-of-the-art query strategies, including some which can leverage the GPU. Clearly defined interfaces allow to combine a multitude of such query strategies with different classifiers, thereby facilitating a quick mix and match, and enabling a rapid development of both active learning experiments and applications. To make various classifiers accessible in a consistent way, it integrates several well-known machine learning libraries, namely, scikit-learn, PyTorch, and huggingface transformers—for which the latter integrations are available as optionally installable extensions. The library is available under the MIT License at https://github.com/webis-de/small-text.

Keywords: active learning, text classification, query strategies, transformers, Python

1. Introduction

Text classification—in the same way as most contemporary machine learning approaches—requires large amounts of data to achieve top performance. However, in most real-world use cases, labeled data does not exist and is expensive to obtain. Active learning (Lewis and Gale, 1994) solves this situation by repeatedly selecting unlabeled data, deemed to be informative according to a so-called query strategy. Subsequently, those samples are labeled by a human annotator, a new model is trained on all data labeled so far, and the process is repeated. Active learning aims at minimizing the amount of labeled data required, while maximizing the resulting model’s performance, e.g., text classification accuracy.

Identifying and reusing code patterns is at the heart of software development, but also essential to science, especially in the fields of machine learning, where most experiments involve software to some extent. Moreover, in computer science, and especially in machine learning, demands for reproducible experiments are rising, which is why an increasing number of publications include a public code repository. However, many of these repositories reimplement existing approaches in their experiments. This is unfortunate, since, as described by Sonnenburg et al. (2007), there are many advantages of using existing software...
solutions in machine learning, among others, the faster detection of bugs, reduced implementation time, and improved reproducibility of experiment code. From a researcher’s point of view the most important factor among those advantages is the reduced risk of having unspotted bugs—since a single unnoticed bug could invalidate the whole experiment. Second to that, the reduced effort of reusing existing components is critical and makes complex experiments possible which would otherwise have been infeasible. This particularly affects active learning, which typically consists of a classifier, a query strategy, and optionally a stopping criterion. Due to this modular nature, single components are mostly interchangeable, which results in many possible combinations and likewise in numerous lines of code. By implication, applied active learning necessarily involves a lot of code, and therefore it can benefit greatly from reusing existing open source software. Although there are existing solutions for general active learning, only few consider text classification, which requires additional functionality such as word embeddings (Mikolov et al., 2013) or languages models (Devlin et al., 2019). To fill this gap, we present a library, which provides robust components for building active learning experiments or applications for text classification.

2. Overview

The main goal of small-text\textsuperscript{1} is to offer state-of-the-art active learning for text classification in a convenient and robust way for both research and applications. For this purpose, we built a modular pool-based active learning mechanism, exposing interfaces for classifiers and query strategies. The core of small-text integrates scikit-learn (Pedregosa et al., 2011), which makes it possible to directly use scikit-learn’s classifiers for active-learning. By providing the means to use many classifiers and query strategies within this setting, we can easily build active learning setups. We also implemented numerous query strategies, including such, which are only usable on text data (see Section 3). Furthermore, we provided integrations for two well-known libraries, namely PyTorch (Paszke et al., 2019) and transformers (Wolf et al., 2020), through which it is possible to easily use CUDA-based GPU computing and transformer models respectively. The architecture is again organized

\textsuperscript{1} Version 1.0.0a3 at the time of writing.
modular as shown in Figure 1, so that both integrations are completely optional, resulting in a slim core which can be used without unnecessary dependencies in a CPU-only scenario.

The library is available via the python packaging index and can be installed with just a single command via `pip install small-text`, and similarly, integrations can be enabled using the optional requirements arguments of Python’s setuptools, e.g., the `transformers` integration is installed using `pip install small-text[transformers]`. The robustness of the implementation is secured by unit and integration tests. Detailed examples, an API documentation, and common usage patterns are available in the online documentation.3

### 3. Query Strategies

The most important component of an active learning setup is the query strategy, which selects the instances to be labeled. We offer a wide range of query strategies: (i) confidence-based strategies: least confidence (Lewis and Gale, 1994), prediction entropy (Roy and McCallum, 2001), breaking ties (Luo et al., 2005); (ii) embedding-based strategies: BADGE (Ash et al., 2020), BERT K-Means (Yuan et al., 2020); and (iii) expected gradient length strategies: EGL (Settles et al., 2007), EGL-max (Zhang et al., 2017). Since there is an abundance of query strategies this list will likely never be exhaustive, and moreover, strategies from other domains such as computer vision are not always applicable to the text domain (e.g., Konyushkova et al., 2015).

### 4. Comparison to Previous Software

Unsurprisingly, with active learning having been researched and applied for decades, there are numerous other libraries available. For brevity’s sake, we selected the most relevant—in the sense of projects having either a related publication or a larger user base—open-source projects, which are listed in Table 1 and discussed in the following.

<table>
<thead>
<tr>
<th>Name</th>
<th>Language</th>
<th>License</th>
<th>Public Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>JCLAL</td>
<td>Java</td>
<td>GPL</td>
<td><a href="https://github.com/ogreyesp/JCLAL">https://github.com/ogreyesp/JCLAL</a></td>
</tr>
<tr>
<td>libact</td>
<td>Python</td>
<td>BSD-2-Clause</td>
<td><a href="https://github.com/ntucllab/libact">https://github.com/ntucllab/libact</a></td>
</tr>
<tr>
<td>ALiPy</td>
<td>Python</td>
<td>BSD-3-Clause</td>
<td><a href="https://github.com/NUAA-AL/ALiPy">https://github.com/NUAA-AL/ALiPy</a></td>
</tr>
<tr>
<td>modAL</td>
<td>Python</td>
<td>MIT</td>
<td><a href="https://github.com/modAL-python/modAL">https://github.com/modAL-python/modAL</a></td>
</tr>
<tr>
<td>lrtc</td>
<td>Python</td>
<td>Apache 2.0</td>
<td><a href="https://github.com/IBM/low-resource-text-classification-framework">https://github.com/IBM/low-resource-text-classification-framework</a></td>
</tr>
<tr>
<td>small-text</td>
<td>Python</td>
<td>MIT</td>
<td><a href="https://github.com/webis-de/small-text">https://github.com/webis-de/small-text</a></td>
</tr>
</tbody>
</table>

Table 1: A descriptive overview of small-text and similar active learning libraries.

JCLAL (Reyes et al., 2016) is a generic framework for active learning which is implemented in Java and can be used either through XML configurations or directly from the code. Besides numerous single- and multi-label query strategies, it offers an experimental setting which, among others, shows a graphical report of the learning curve. The aim of libact (Yang et al., 2017) is to provide a convenient way to use active-learning in real-world ap-

2. https://small-text.readthedocs.io
lications. Notably, it includes a well-known meta-learning strategy (Hsu and Lin, 2015) as well as cost-sensitive active learning. Finally, modAL (Danka and Horvath, 2018) offers a broad range of active learning functionality, including regression, multi-label classification and stream-based active learning, as well as a multitude of query strategies. It also builds on scikit-learn by default, and provides instructions how to include GPU-based models using Keras and PyTorch. ALiPy (Tang et al., 2019) provides an active learning framework targeted at the experimental active learning setting. Apart from providing more than 20 query strategies, it supports extended active learning settings, e.g., active learning with noisy annotators. The lrtc (low resource text classification) framework (Ein-Dor et al., 2020) also focuses on text classification and has a number of built-in models, datasets, and query strategies to perform experiments on standard text classification benchmarks. It also provides a pre-implemented BERT classifier using the transformers library. Being focused on text classification, it is the most similar to our library, however, it is correctly called a framework, bringing an experimental frame, which is easily adaptable to other scenarios.

All of the listed libraries claim modularity and/or extensibility, however, this is partly owed to active learning’s conceptual structure (i.e., components like classifiers, query strategies, and stopping criteria) in which the single components are usually independent as it is. While all of them have some overlap, only modAL, lrtc and small-text offer access to GPU-based deep learning frameworks, which is indispensable for text classification (and natural language processing in general as well) due to the recent success and ubiquity of transformer-based models (Vaswani et al., 2017; Devlin et al., 2019). Only lrtc and small-text focus on text classification, but they differ conceptually in so far that lrtc is a framework offering a set of procedures as a frame whose parts can be adapted, while small-text, a library, offers a tool box of components which can be used to build larger components or applications. Moreover, solely small-text offers the CNN-based gradient strategies of Zhang et al. (2017), which are unique to text classification and an important baseline for active-learning-based text classification. Most importantly, the transformers integration makes any transformers-compatible model usable in a very convenient way, thereby enabling state-of-the-art active learning for text classification in a few lines of code.

5. Conclusion

We introduced small-text, a modular Python library, which offers active learning for text classification. It integrates existing libraries like scikit-learn, PyTorch, and transformers, which makes state-of-the-art active learning easily accessible to the Python ecosystem. Due to its adherence to software engineering best practices, it provides a robust set of components to quickly apply active learning for text classification in both experiments and applications.

Acknowledgments

This research was partially funded by the Development Bank of Saxony (SAB) under project numbers 100335729 and 100400221.
References


Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam

