Data versioning in machine-learning architecture

Peter Bartoš, Stanislav Krištof

Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava

September 29, 2024

Abstract

Data versioning plays a crucial role in modern machine learning architecture, ensuring that the complex and ever-evolving datasets that provide the basis for models can be tracked, compared, and managed efficiently. At its core, data versioning refers to the practice of creating unique references for different states of a dataset over time, allowing us to trace changes, restore previous versions, and debug issues. This is vital in machine learning workflows, where even small changes in data can significantly impact model performance.

In this domain, data versioning supports reproducibility by maintaining a consistent link between datasets and the models trained on them. Without version control, it becomes challenging to recreate experiments, leading to inconsistencies in predictions and hindering model audits. Versioning also simplifies collaboration across teams, enabling multiple stakeholders to work on the same data without overwriting each other's progress.

Basic approaches to data versioning include full duplication of datasets, where copies are saved with each change, and metadata-based versioning, where timestamps indicate the validity of each record. Advanced solutions (like lakeFS and DVC) deal with versioning as a core component of machine learning architecture. They enable storage-efficient data commits, branching, and comparison, similar to how Git handles version control in software development.

Overall, data versioning enhances productivity, reduces errors, and fosters an engineering-driven approach to handling data in machine learning pipelines, ultimately enabling smoother transitions between development stages and more robust model deployment. The objective of the project is to go over these systems and provide detailed overviews of how data versioning has such a crucial role in machine learning architecture.

1 Introduction

...

The rest of this report[1] is structured as follows. Section 2 provides an insight into... Section 3 explains in more detail some important approaches to... Section 4 brings the initial steps to... (Characterize each section by a sentence.) Section 8 concludes the paper and indicates some directions for further work.

2 Insight into...

Present your insight into the state of the art. Favor comparison and critique over description. Avoid lengthy descriptions with lots of quoted material.

Refer to literature properly, e.g., "Many authors have tried to... [**?**, **?**], but..."

Use your own title for this section.

You may structure your sections (see below). If you use subsections, use at least two, i.e., don't put only one subsection.

It might be a good idea to explain the structure of the rest of the section. Sometimes, an explicit way of doing at just as is demonstrated at the end of the introduction (see Section 1).

2.1 Some Aspects

...

2.2 Other Aspects

...

3 Important Approaches to...

You may need one more section to treat the state of the art. Everything said for the previous section, holds for this one, too.

4 Initial Steps to...

Describe your own approach. Of course, use your own title for this section.

Put your diagrams in figures as so-called floating object. Refer to them using their numbers. E.g., "in Figure 1 we can see..."

You may include code snippets to explain what you've done:

[1]This report has been submitted in partial fulfillemnt of the Aspect-Oriented Software Development 2018/19 course completion conditions (`http://fiit.sk/~vranic/aosd/`). Supervised by Valentino Vranić.

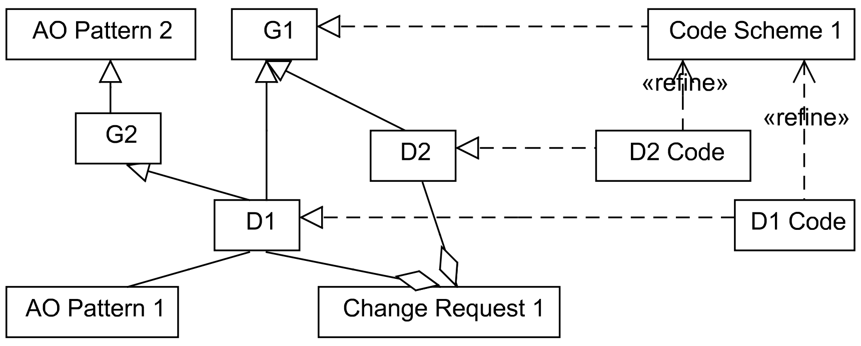

Figure 1: Generally applicable and domain specific changes.

```
public class SMTPServerM extends SMTPServer {
   . . .
}
. . .
public aspect SMTPServerBackupA {
   public pointcut SMTPServerConstructor(URL url, String user, String password):
      call(SMTPServer.new(..)) && args (url, user, password);
   SMTPServer around(URL url, String user, String password):
      SMTPServerConstructor(url, user, password) {
      return getSMTPServerBackup(proceed(url, user, password));
   }
   private SMTPServer getSMTPServerBackup(SMTPServer obj) {
      if (obj.isConnected()) {
         return obj;
      }
      else {
         return new SMTPServerM(obj.getUrl(), obj.getUser(),
            obj.getPassword());
      }
   }
}
```

If you need to display more code, use appendices referring the reader to them, e.g., "see Appendix A for a more detailed example."

5 Further Steps to...

You may need several sections to describe your approach.

6 Evaluation

You may describe your evaluation efforts in a separate, often generically entitled section.

6.1 Essential Evaluation

Use your own title here.

6.2 Threats to Validity

...

7 Related Work

Compare your achievements to related ones achieved by others.

8 Conclusions and Further Work

Emphasize the main results.

Indicate what can be done next.

The concluding section is typically not decomposed into subsections. Simply use several paragraphs to present conclusions, and then use at least one paragraph to indicate further/future work.

References

A Some Appendix

B Yet Another Appendix