Caching

Austin Dunn, Samuel Huang, Benson Chau

Overview

1. What is Caching
2. How to use the Caching
3. Statistics on Feat. Data
4. The Future of EMADE Data Processing

Caching: What is it?

Wikipedia Definition: a **cache** is a hardware or software component that stores data so that future requests for that data can be served faster; the data stored in a cache might be the result of an earlier computation or a copy of data stored elsewhere.

In our case with EMADE, we want to store the time consuming computation on the disk, and read it when we need to use it.

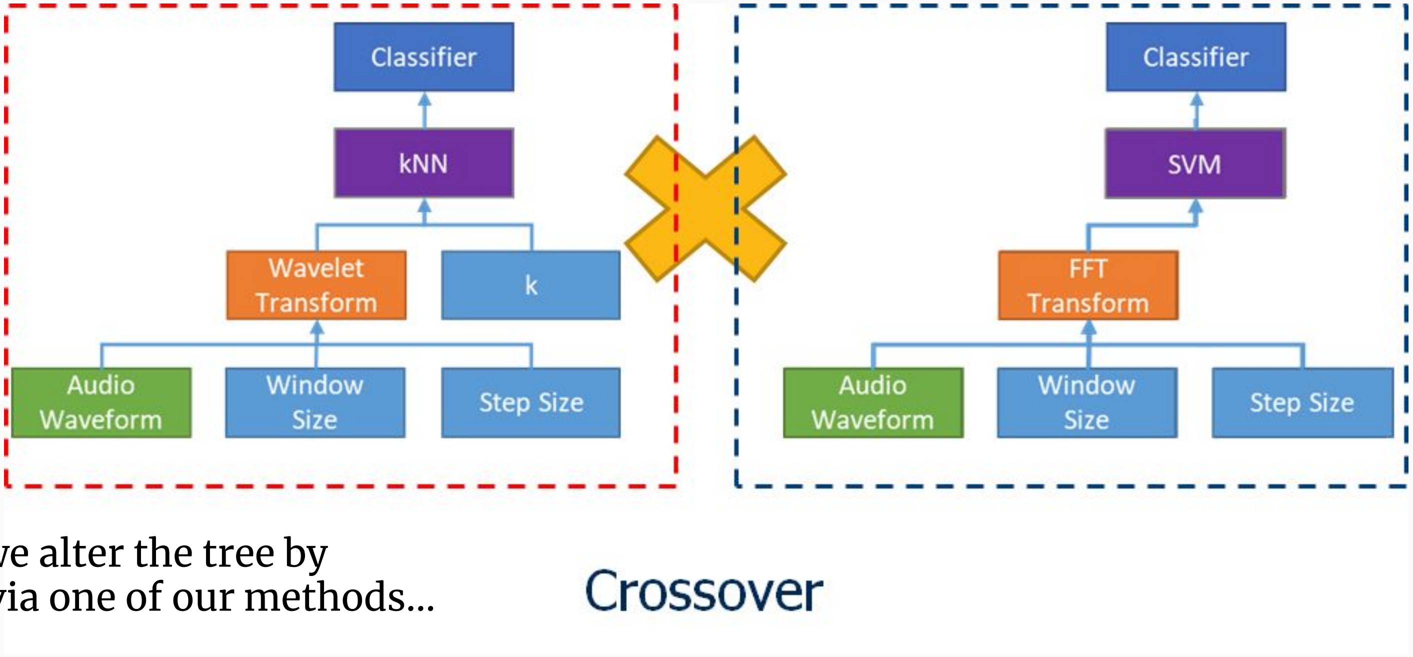
Caching: Why?

Caching is critical to EMADE as EMADE includes a number of **very expensive** computations that could save both run time and computational power.

Thus, **reusing the data** that we've worked so hard to get inherently makes sense.

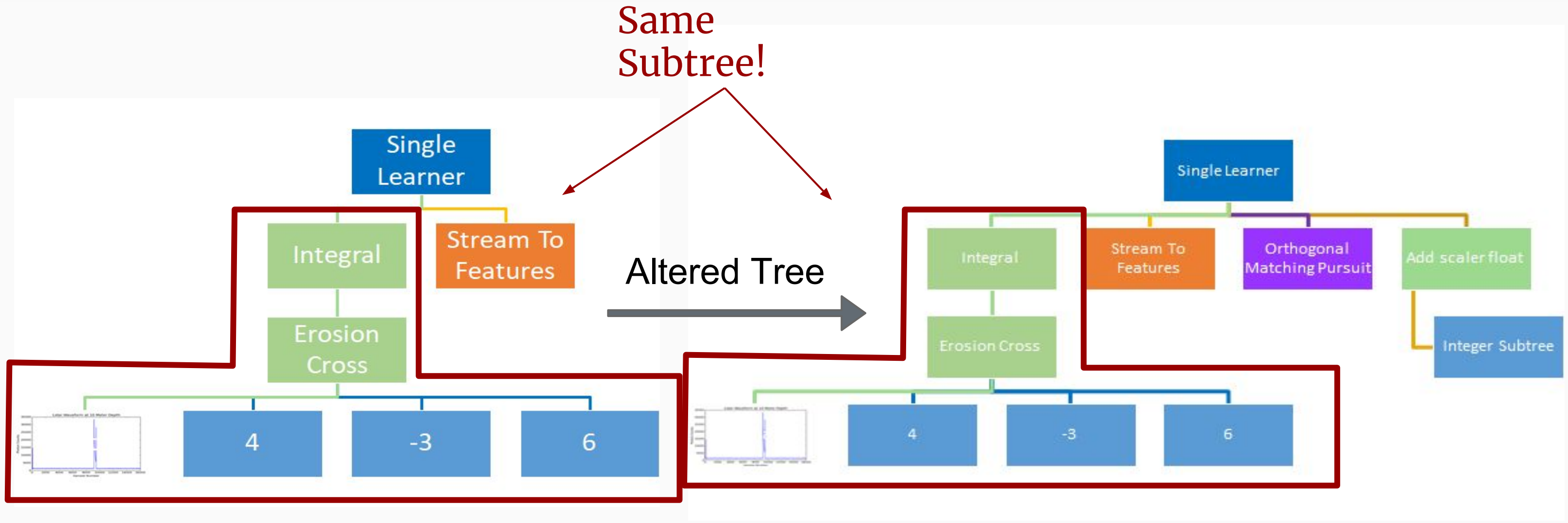
As EMADE is updated to include Deep Learning frameworks and other more expensive computations, we could potentially save **hundreds of hours of training time** of these computationally heavy neural networks.

Caching: How it works



In Emade, we alter the tree by evolving it via one of our methods...

Caching: How it works



But we might alter the tree in a way such that the subtree is the same, and only the top of the tree is changed. In that case, we want to avoid expensive computation and use the cache for these subtrees.

Caching: How it works

How do we identity data stored in the cache?

With a **unique key**!

Method Name + Parameters + Hash of the Previous Data

Caching: How it works

What happens when **the cache is full**?

At the end of every gen, we optimize the cache to store data by this metric:

# of times Key was referenced * Time taken to create Data

Statistics: Dota Dataset

- 92645 examples
- 116 features
- Sparse (Input Data is binary 0/1 consisting mostly of 0s)
- **Low Processing Time**
- **Large File Size**

Statistics: Titanic Dataset

- 891 examples
- 11 features
- **Dense** (Features consist of multiple data types and float ranges)
- **High Processing Time**
- **Small File Size**

Statistics: Compare & Contrast

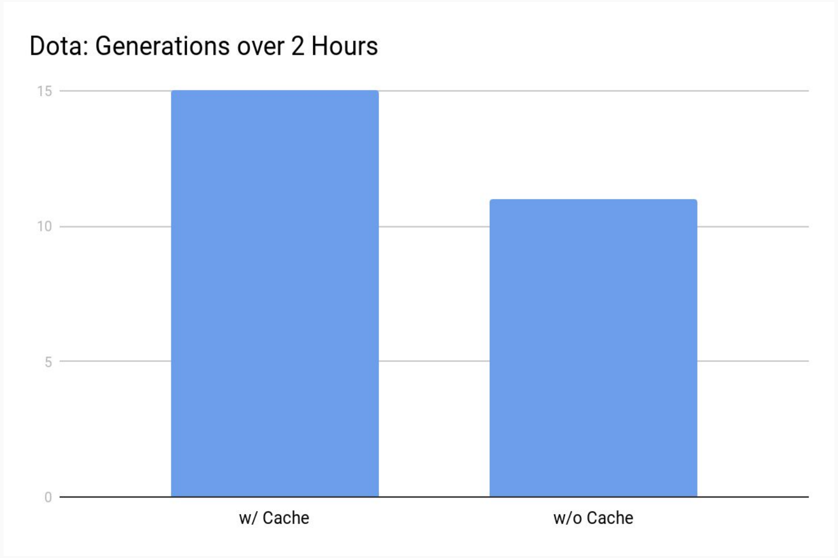
- Both Datasets ran 120 minute trials
- 2 Trials for each Dataset, 1 with Caching, 1 without Caching
- Showcase both small and large file sizes
- Both these datasets do not benefit from caching on paper
- The dataset either runs quickly (Dota) or already uses little space (Titanic)

Statistics: Dota Trial

Over 120 minutes, EMADE would have spent **59.8 minutes** on valid primitives

With Caching instead, EMADE only spent **29.9 minutes** on valid primitives

We **Halved** the Processing Time!

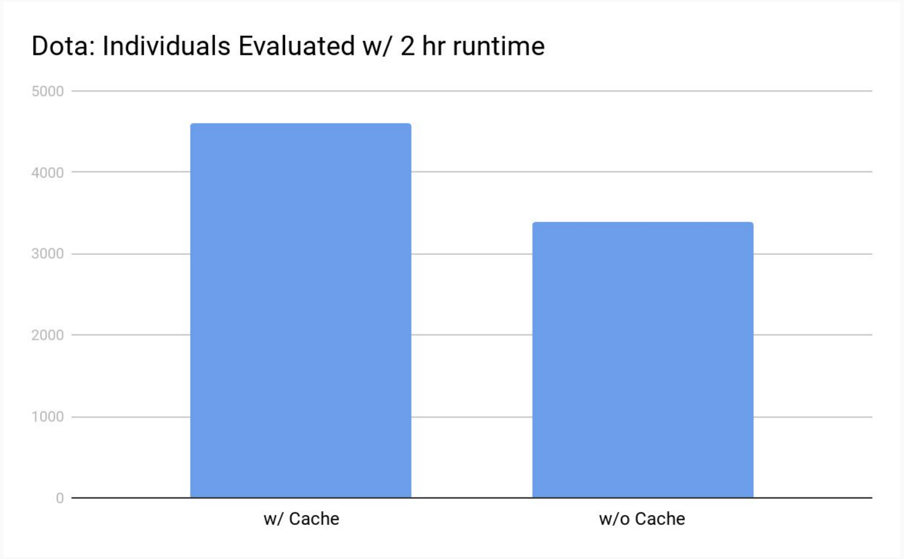


Statistics: Dota Trial

36% more individuals were evaluated w/ caching in 2 hours.

This speeds up the evolutionary process of EMADE!

Over 2 hours we also stored **1.2GB** of re-usable data

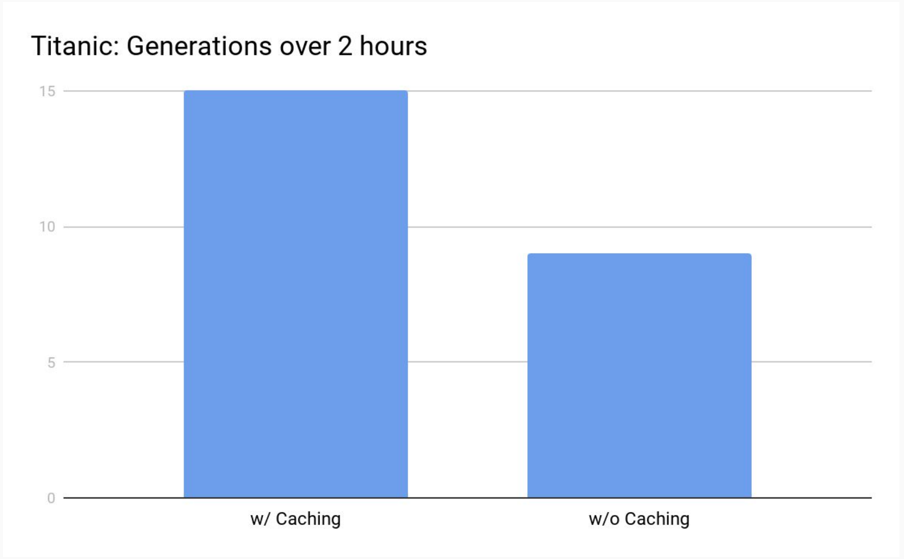


Statistics: Titanic Trial

Over 120 minutes, EMADE would have spent **5.9 minutes** on valid primitives

With Caching instead, EMADE only spent **35 seconds** on valid primitives

This is **90.11% Efficiency** on Stored Data

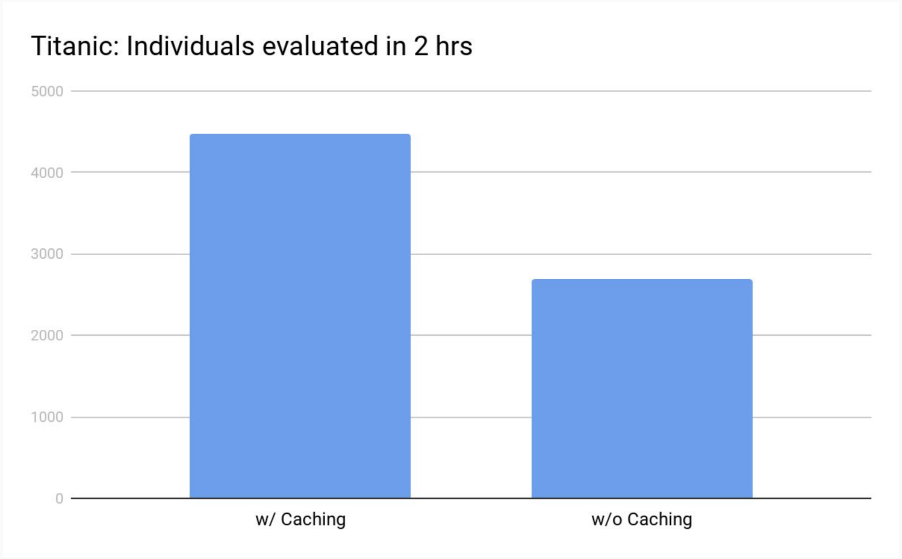


Statistics: Titanic Trial

66% more individuals were evaluated w/ caching in 2 hours.

This is **30% higher** than the previous dataset!

Over 2 hours we also stored **390 kB** of re-usable data



Statistics: Compression Tradeoff

- Compressed data uses **56.78% less space**.

- Compressed data uses **95.59% higher write time**.

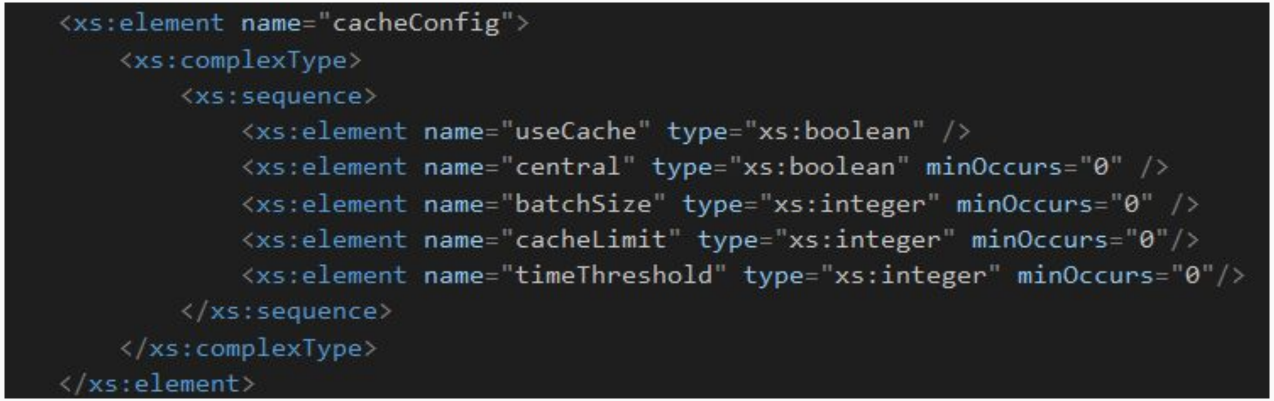
- Using an SSD can save write time when the dataset is large

* Compression trade-offs shown here do not take into account read time of compressed data.

How to enable caching for EMADE

Cache Parameters

- useCache (T/F)
- Central (T/F)
- Cache Limit (KB)
- timeThreshold (s)



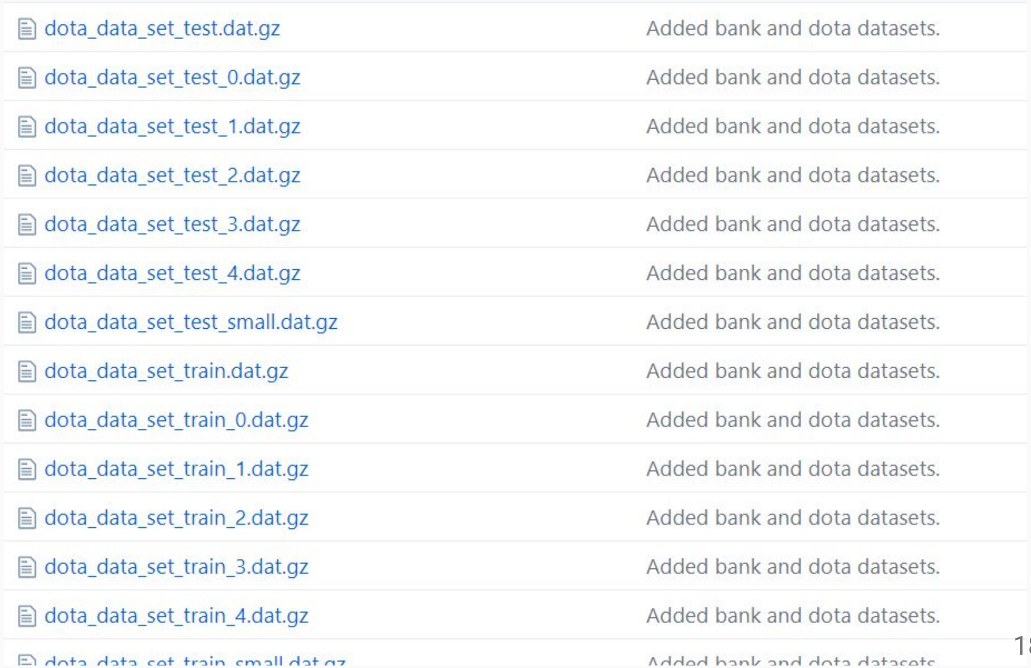
```
<xs:element name="cacheConfig">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="useCache" type="xs:boolean" />
            <xs:element name="central" type="xs:boolean" minOccurs="0" />
            <xs:element name="batchSize" type="xs:integer" minOccurs="0" />
            <xs:element name="cacheLimit" type="xs:integer" minOccurs="0"/>
            <xs:element name="timeThreshold" type="xs:integer" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

How to enable caching for EMADE

Directories

```
<trainFilename>datasets/sample/sample_train_0.dat.gz</trainFilename>
```

- MAKE SURE YOU INCLUDE THE FOLD #
- Fold: “train0”
- Base Directory: “datasets/sample/”
- Gen Directory: “datasets/sample/gen_data”



dota_data_set_test.dat.gz	Added bank and dota datasets.
dota_data_set_test_0.dat.gz	Added bank and dota datasets.
dota_data_set_test_1.dat.gz	Added bank and dota datasets.
dota_data_set_test_2.dat.gz	Added bank and dota datasets.
dota_data_set_test_3.dat.gz	Added bank and dota datasets.
dota_data_set_test_4.dat.gz	Added bank and dota datasets.
dota_data_set_test_small.dat.gz	Added bank and dota datasets.
dota_data_set_train.dat.gz	Added bank and dota datasets.
dota_data_set_train_0.dat.gz	Added bank and dota datasets.
dota_data_set_train_1.dat.gz	Added bank and dota datasets.
dota_data_set_train_2.dat.gz	Added bank and dota datasets.
dota_data_set_train_3.dat.gz	Added bank and dota datasets.
dota_data_set_train_4.dat.gz	Added bank and dota datasets.
dota_data_set_train_small.dat.gz	Added bank and dota datasets

Future Work: Known Issues

- Hypervolume calculation occasionally crashes master process
- Stream Data supported but currently not saved by cache
- Feature Selection methods currently disabled when using caching (these methods damage data object integrity)
- Stored data is not removed when EMADE is finished running

Future Work: Additional Features

- Support saving and loading of Stream Data
- Add Data Compression Options
- Integrate Feature Selection into Caching Framework
- Add methods for clearing all old cache data at the start of EMADE