## LIRS: Enabling efficient machine learning on NVM-based storage via a lightweight implementation of random shuffling

由于全局shuffle对于提高测试的准确度并减少训练时间，所以本文针对SVM与DNN中的数据shuffle进行了研究，并使用Optane SSD替代原始的HDD并引入了KV的思想对数据集进行处理

解决问题：

1 无法全局shuffle

2 随机读太慢

3 遇到稀疏格式的文件不好处理

4 一个dataset文件太小，导致浪费的问题

优点如下：

1 使用KV的思想将data进行kv操作，生成Key的table表存入memory中可以做到全局shuffle（原始是将数据直接存入memory，并不能全部存下所以需要分batch存储）

这样可以减少训练轮数，并提高效率。

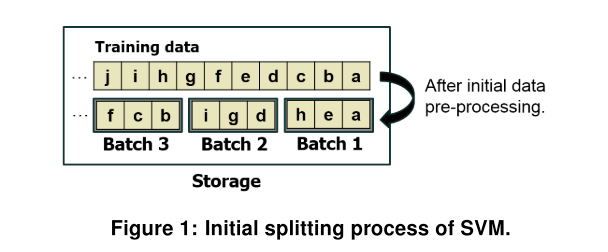
2 使用了Inter Optane SSD来提升随机寻找数据的效率

3 设计Data Format Aware Location Generator与Page-aware Random Shufflin来解决稀疏数据格式的问题以及小训练量数据的问题

### Motivation

随机shuffle工作在现在的系统中并不是直接进行，并且在HDD中非常的慢，所以准备使用新的方法与新的SSD介质进行。

- SVM中现有的方法为Block Minimization Framework (BMF)，该方法先将数据读入memory中然后再选择所需训练的batch

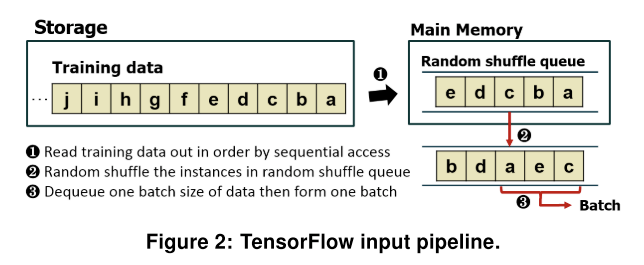


不足之处：

1 由于HDD随机写的时间太长，所以IO时间太长

2 每一个epoch中的数据顺序不改变，所以会使得收敛速率降低

- DNN使用了管道技术



不足之处：

1 shuffle全部的数据集需要与操作

2 随机化程度受到memory的制约

### 使用SSD进行随机化操作

本文设计了LIRS：The core concept of LIRS is to randomly assign the training instances to each different batches on the host side to achieve the random shuffling effect.

在memory中维护了一个Key table，记录了数据集的信息。使用该table表查找对应的data位置并取数据

挑战

1 需要知道数据的位置，而数据可以存储为稀疏格式与非稀疏格式。非稀疏格式直接读取，而非稀疏格式需要特殊处理。

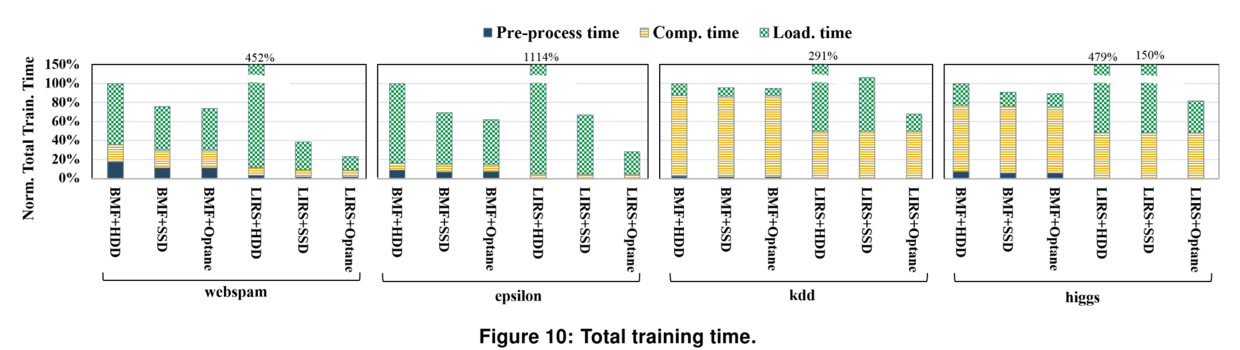
2 当数据非常小的时候，有可能OS的虚拟页可以装下多个数据，此时则会多读取许多内容，导致效率降低

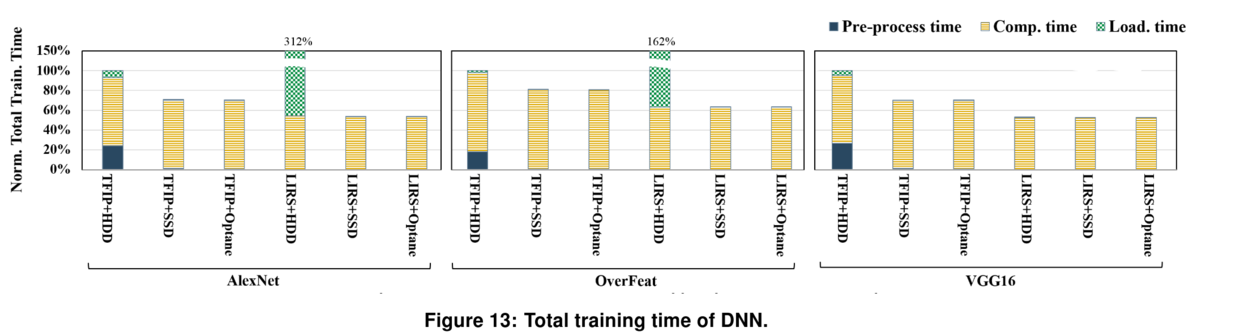
为了解决这两个问题使用了两种方法：

- Data Format Aware Location Generator

- Page-aware Random Shuffling

最后对LIRS方法做了评估：

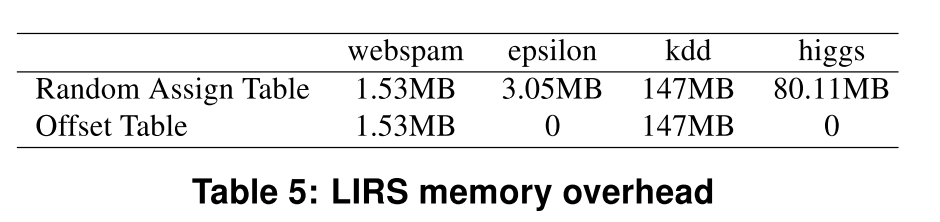




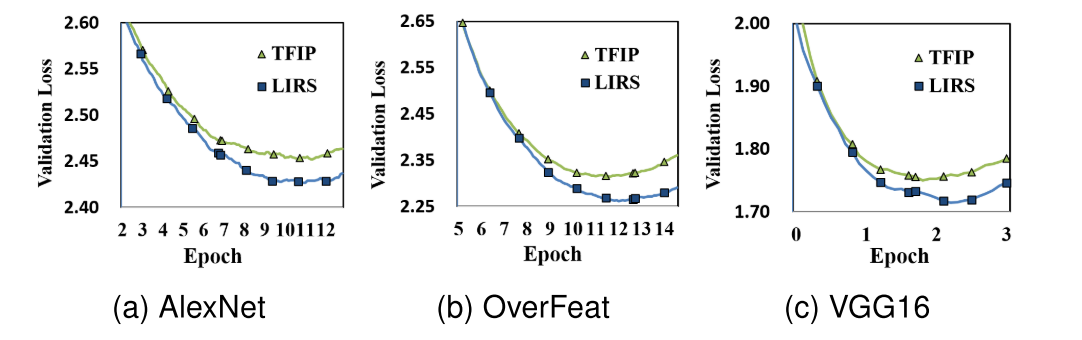
最后的结论为：

- SVM： LIRS converges faster than BMF at all of the four training datasets

对于额外的数据表来说：LIRS introduces less than 1% memory space overhead for webspam and epsilon in a 1GB main memory.



- DNN



对于额外数据表：

LIRS needs 9.8MB (< 0.1%) additional memory space to store the random assignment table

LIRS can save a large amount of CPU memory space：LIRS可以节省很大一部分memory空间以供CPU高效运行