1 Flinklink: Code Examples

- Input/output
  - Reading data
  - Write data
- Parametric Learning
  - Parallel Maximum Likelihood
  - Distributed Variational Message Pasing
  - Distributed VI
  - Stochastic VI
- Extensions and applications
  - Latent variable models with Flink
  - Concept drift

1.1 Input/output

1.1.1 Reading data

In this example we show how can we read a dataset using Flink. Note that the process is the same regardless being a single or a distributed file.

```
1  package eu.amidst.flinklink.examples.io;
2
3  import eu.amidst.core.datastream.DataInstance;
4  import eu.amidst.flinklink.core.data.DataFlink;
5  import eu.amidst.flinklink.core.io.DataFlinkLoader;
6  import org.apache.flink.api.java.ExecutionEnvironment;
7
8  /**
9   * Created by rcabanas on 10/06/16.
10  */
11 public class DataStreamLoaderExample {
12     public static void main(String[] args) throws Exception {
13
14         //Set the environment variable
15         final ExecutionEnvironment env = ExecutionEnvironment.
               getExecutionEnvironment();
16
17         //Paths to datasets
18         String simpleFile = "datasets/simulated/syntheticData.arff";
19         String distriFile = "datasets/simulated/distributed.arff";
20
21         //Load the data
22         DataFlink<DataInstance> dataSimple = DataFlinkLoader.open(
               env, simpleFile, false);
```

```
23          DataFlink<DataInstance> dataDistri = DataFlinkLoader.open(
                env,distriFile, false);
24
25          //Print the data
26          dataSimple.getDataSet().print();
27          dataDistri.getDataSet().print();
28
29      }
30  }
```

[Back to Top]

1.1.2 Writing data

Below we generate a random Flink dataset with 1000 instances, 2 discrete variables and 3 continuous ones. The seed used is 1234. Eventually, we save it as a distributed dataset (format ARFF folder).

```
1   package eu.amidst.flinklink.examples.io;
2
3   import eu.amidst.core.datastream.DataInstance;
4   import eu.amidst.flinklink.core.data.DataFlink;
5   import eu.amidst.flinklink.core.io.DataFlinkWriter;
6   import eu.amidst.flinklink.core.utils.DataSetGenerator;
7
8   /**
9    * Created by rcabanas on 09/06/16.
10   */
11  public class DataStreamWriterExample {
12      public static void main(String[] args) throws Exception {
13
14          //generate a random dataset
15          DataFlink<DataInstance> dataFlink = new DataSetGenerator().
                generate(1234,1000,2,3);
16
17          //Saves it as a distributed arff file
18          DataFlinkWriter.writeDataToARFFFolder(dataFlink, "datasets/
                simulated/distributed.arff");
19      }
20  }
21
22
23  //TODO: Write to standard arff -->> convert to datastream??
```

[Back to Top]

1.2 Parametric learning

Here give examples of the provided algorithms by AMiDST for learning the probability distributions from a Flink data set. For shake of simplicity, we will

consider the Naive Bayes DAG structure. Note that the code is almost the same of each of the algoritms, they only differ on the constructor used (e.g. *new ParallelMaximumLikelihood()*, *new dVMP()*, etc.)

1.2.1 Parallel Maximum Likelihood

```
1  package eu.amidst.flinklink.examples.learning;
2
3  import eu.amidst.core.datastream.DataInstance;
4  import eu.amidst.core.models.BayesianNetwork;
5  import eu.amidst.core.models.DAG;
6  import eu.amidst.core.utils.DAGGenerator;
7  import eu.amidst.flinklink.core.data.DataFlink;
8  import eu.amidst.flinklink.core.learning.parametric.
       ParallelMaximumLikelihood;
9  import eu.amidst.flinklink.core.learning.parametric.
       ParameterLearningAlgorithm;
10 import eu.amidst.flinklink.core.utils.DataSetGenerator;
11
12 /**
13  * Created by rcabanas on 14/06/16.
14  */
15 public class ParallelMLExample {
16     public static void main(String[] args) throws Exception {
17
18         //generate a random dataset
19         DataFlink<DataInstance> dataFlink = new DataSetGenerator().
               generate(1234,1000,5,0);
20
21         //Creates a DAG with the NaiveBayes structure for the random
               dataset
22         DAG dag = DAGGenerator.getNaiveBayesStructure(dataFlink.
               getAttributes(), "DiscreteVar4");
23         System.out.println(dag.toString());
24
25
26         //Create the Learner object
27         ParameterLearningAlgorithm learningAlgorithmFlink =
28                 new ParallelMaximumLikelihood();
29
30         //Learning parameters
31         learningAlgorithmFlink.setBatchSize(10);
32         learningAlgorithmFlink.setDAG(dag);
33
34         //Initialize the learning process
35         learningAlgorithmFlink.initLearning();
36
37         //Learn from the flink data
38         learningAlgorithmFlink.updateModel(dataFlink);
39
```

```
40          //Print the learnt BN
41          BayesianNetwork bn = learningAlgorithmFlink.
                getLearntBayesianNetwork();
42          System.out.println(bn);
43
44
45
46      }
47  }
```

[Back to Top]

1.2.2 Distributed Variational Message Passing

```
1   package eu.amidst.flinklink.examples.learning;
2
3   import eu.amidst.core.datastream.DataInstance;
4   import eu.amidst.core.models.BayesianNetwork;
5   import eu.amidst.core.models.DAG;
6   import eu.amidst.core.utils.DAGGenerator;
7   import eu.amidst.flinklink.core.data.DataFlink;
8   import eu.amidst.flinklink.core.learning.parametric.
        ParameterLearningAlgorithm;
9   import eu.amidst.flinklink.core.learning.parametric.dVMP;
10  import eu.amidst.flinklink.core.utils.DataSetGenerator;
11
12  /**
13   * Created by rcabanas on 14/06/16.
14   */
15  public class dVMPExample {
16      public static void main(String[] args) throws Exception {
17
18          //generate a random dataset
19          DataFlink<DataInstance> dataFlink = new DataSetGenerator().
                generate(1234,1000,5,0);
20
21          //Creates a DAG with the NaiveBayes structure for the random
                dataset
22          DAG dag = DAGGenerator.getNaiveBayesStructure(dataFlink.
                getAttributes(), "DiscreteVar4");
23          System.out.println(dag.toString());
24
25
26          //Create the Learner object
27          ParameterLearningAlgorithm learningAlgorithmFlink =
28                  new dVMP();
29
30          //Learning parameters
31          learningAlgorithmFlink.setBatchSize(10);
32          learningAlgorithmFlink.setDAG(dag);
```

```
33
34          //Initialize the learning process
35          learningAlgorithmFlink.initLearning();
36
37          //Learn from the flink data
38          learningAlgorithmFlink.updateModel(dataFlink);
39
40          //Print the learnt BN
41          BayesianNetwork bn = learningAlgorithmFlink.
                getLearntBayesianNetwork();
42          System.out.println(bn);
43
44
45
46      }
47  }
```

[Back to Top]

1.2.3 Distributed VI

```
1   package eu.amidst.flinklink.examples.learning;
2
3   import eu.amidst.core.datastream.DataInstance;
4   import eu.amidst.core.models.BayesianNetwork;
5   import eu.amidst.core.models.DAG;
6   import eu.amidst.core.utils.DAGGenerator;
7   import eu.amidst.flinklink.core.data.DataFlink;
8   import eu.amidst.flinklink.core.learning.parametric.DistributedVI;
9   import eu.amidst.flinklink.core.learning.parametric.
        ParameterLearningAlgorithm;
10  import eu.amidst.flinklink.core.utils.DataSetGenerator;
11
12  /**
13   * Created by rcabanas on 14/06/16.
14   */
15  public class DistributedVIExample {
16      public static void main(String[] args) throws Exception {
17
18          //generate a random dataset
19          DataFlink<DataInstance> dataFlink = new DataSetGenerator().
                generate(1234,1000,5,0);
20
21          //Creates a DAG with the NaiveBayes structure for the random
                dataset
22          DAG dag = DAGGenerator.getNaiveBayesStructure(dataFlink.
                getAttributes(), "DiscreteVar4");
23          System.out.println(dag.toString());
24
25
```

```
26          //Create the Learner object
27          ParameterLearningAlgorithm learningAlgorithmFlink =
28                  new DistributedVI();
29
30          //Learning parameters
31          learningAlgorithmFlink.setBatchSize(10);
32          learningAlgorithmFlink.setDAG(dag);
33
34          //Initialize the learning process
35          learningAlgorithmFlink.initLearning();
36
37          //Learn from the flink data
38          learningAlgorithmFlink.updateModel(dataFlink);
39
40          //Print the learnt BN
41          BayesianNetwork bn = learningAlgorithmFlink.
                getLearntBayesianNetwork();
42          System.out.println(bn);
43
44
45
46      }
47  }
```

[Back to Top]

1.2.4 Stochastic VI

An example of the learning algorithm Stochastic VI is given below. Note that two specific parameters must be set, namely the *learning factor* and the *data size*.

```
1   package eu.amidst.flinklink.examples.learning;
2
3   import eu.amidst.core.datastream.DataInstance;
4   import eu.amidst.core.models.BayesianNetwork;
5   import eu.amidst.core.models.DAG;
6   import eu.amidst.core.utils.DAGGenerator;
7   import eu.amidst.flinklink.core.data.DataFlink;
8   import eu.amidst.flinklink.core.learning.parametric.
        ParameterLearningAlgorithm;
9   import eu.amidst.flinklink.core.learning.parametric.StochasticVI;
10  import eu.amidst.flinklink.core.utils.DataSetGenerator;
11
12  /**
13   * Created by rcabanas on 14/06/16.
14   */
15  public class StochasticVIExample {
16      public static void main(String[] args) throws Exception {
17
18          //generate a random dataset
```

```
19          DataFlink<DataInstance> dataFlink = new DataSetGenerator().
                generate(1234,1000,5,0);
20
21          //Creates a DAG with the NaiveBayes structure for the random
                dataset
22          DAG dag = DAGGenerator.getNaiveBayesStructure(dataFlink.
                getAttributes(), "DiscreteVar4");
23          System.out.println(dag.toString());
24
25
26          //Create the Learner object
27          ParameterLearningAlgorithm learningAlgorithmFlink =
28                  new StochasticVI();
29
30          //Learning parameters
31          learningAlgorithmFlink.setBatchSize(10);
32          learningAlgorithmFlink.setDAG(dag);
33
34          //Initialize the learning process
35          learningAlgorithmFlink.initLearning();
36
37          //Learn from the flink data
38          learningAlgorithmFlink.updateModel(dataFlink);
39
40          //Specific parameters for the algorithm
41          ((StochasticVI)learningAlgorithmFlink).setLearningFactor(0.7);
42          ((StochasticVI)learningAlgorithmFlink).setDataSetSize(dataFlink.
                getDataSet().count());
43
44
45          //Print the learnt BN
46          BayesianNetwork bn = learningAlgorithmFlink.
                getLearntBayesianNetwork();
47          System.out.println(bn);
48
49
50
51      }
52  }
```

[Back to Top]

1.3 Extensions and applications

1.3.1 Latent variable models with Flink

The module *latent-variable-models* contains a large set of classes that allow to easily learn some of the standard models with latent variables. These models can be learnt from not only from local datasets (e.g. a single ARFF file) but also from distributed ones (e.g. ARFF folder). These last ones are managed

using Flink. In code example shown below the model *Factor Analysis* is learnt from a distributed dataset.

```
1  package eu.amidst.flinklink.examples.extensions;
2
3  import eu.amidst.core.datastream.DataInstance;
4  import eu.amidst.core.models.BayesianNetwork;
5  import eu.amidst.flinklink.core.data.DataFlink;
6  import eu.amidst.flinklink.core.io.DataFlinkLoader;
7  import eu.amidst.latentvariablemodels.staticmodels.FactorAnalysis;
8  import eu.amidst.latentvariablemodels.staticmodels.Model;
9  import org.apache.flink.api.java.ExecutionEnvironment;
10
11 import java.io.FileNotFoundException;
12
13 /**
14  * Created by rcabanas on 14/06/16.
15  */
16 public class LatentModelsFlink {
17     public static void main(String[] args) throws FileNotFoundException {
18
19         //Load the datastream
20         String filename = "datasets/simulated/exampleDS_d0_c5.arff";
21         final ExecutionEnvironment env = ExecutionEnvironment.
               getExecutionEnvironment();
22         DataFlink<DataInstance> data = DataFlinkLoader.
               loadDataFromFile(env, filename, false);
23
24         //Learn the model
25         Model model = new FactorAnalysis(data.getAttributes());
26         ((FactorAnalysis)model).setNumberOfLatentVariables(3);
27         model.updateModel(data);
28         BayesianNetwork bn = model.getModel();
29
30         System.out.println(bn);
31
32     }
33 }
```

[Back to Top]

In previous example,

1.3.2 Concept drift detection

```
1  /*
2   *
3   *
4   * Licensed to the Apache Software Foundation (ASF) under one or more
        contributor license agreements.
```

```
5   * See the NOTICE file distributed with this work for additional
        information regarding copyright ownership.
6   * The ASF licenses this file to You under the Apache License, Version 2.0
        (the "License"); you may not use
7   * this file except in compliance with the License. You may obtain a copy
        of the License at
8   *
9   * http://www.apache.org/licenses/LICENSE-2.0
10  *
11  * Unless required by applicable law or agreed to in writing, software
        distributed under the License is
12  * distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
        CONDITIONS OF ANY KIND, either express or implied.
13  * See the License for the specific language governing permissions and
        limitations under the License.
14  *
15  *
16  */
17
18  package eu.amidst.flinklink.examples.reviewMeeting2015;
19
20  import eu.amidst.core.datastream.DataInstance;
21  import eu.amidst.flinklink.core.conceptdrift.IDAConceptDriftDetector;
22  import eu.amidst.flinklink.core.data.DataFlink;
23  import eu.amidst.flinklink.core.io.DataFlinkLoader;
24  import org.apache.flink.api.java.ExecutionEnvironment;
25
26  /**
27   * Created by ana@cs.aau.dk on 18/01/16.
28   */
29  public class ConceptDriftDetector {
30
31      //public int NSETS = 15;
32
33
34      public static void learnIDAConceptDriftDetector(int NSETS) throws
            Exception {
35          final ExecutionEnvironment env = ExecutionEnvironment.
                getExecutionEnvironment();
36
37          DataFlink<DataInstance> data0 = DataFlinkLoader.
                loadDataFromFolder(env,
38                  "hdfs:///tmp_conceptdrift_data0.arff", false);
39
40
41          long start = System.nanoTime();
42          IDAConceptDriftDetector learn = new IDAConceptDriftDetector()
                ;
43          learn.setBatchSize(1000);
44          learn.setClassIndex(0);
```

```
45          learn.setAttributes(data0.getAttributes());
46          learn.setNumberOfGlobalVars(1);
47          learn.setTransitionVariance(0.1);
48          learn.setSeed(0);
49
50          learn.initLearning();
51          double[] output = new double[NSETS];
52
53          System.out.println("----------------LEARNING DATA
                " + 0 + "----------------------------");
54          double[] out = learn.updateModelWithNewTimeSlice(data0);
55          //System.out.println(learn.getLearntDynamicBayesianNetwork());
56          output[0] = out[0];
57
58          for (int i = 1; i < NSETS; i++) {
59              System.out.println("----------------LEARNING
                    DATA" + i + "
                    ----------------------------");
60              DataFlink<DataInstance> dataNew = DataFlinkLoader.
                    loadDataFromFolder(env,
61                      "hdfs:///tmp_conceptdrift_data" + i + ".arff", false);
62              out = learn.updateModelWithNewTimeSlice(dataNew);
63              //System.out.println(learn.getLearntDynamicBayesianNetwork
                    ());
64              output[i] = out[0];
65
66          }
67          long duration = (System.nanoTime() - start) / 1;
68          double seconds = duration / 1000000000.0;
69
70          System.out.println("Running time" + seconds + " seconds");
71
72          //System.out.println(learn.getLearntDynamicBayesianNetwork());
73
74          for (int i = 0; i < NSETS; i++) {
75              System.out.println("E(H_"+i+") =\t" + output[i]);
76          }
77
78      }
79
80      public static void main(String[] args) throws Exception {
81
82          int NSETS = Integer.parseInt(args[0]);
83
84          learnIDAConceptDriftDetector(NSETS);
85      }
86
87  }
```

[Back to Top]