

AMIDST: a Java Toolbox for Scalable Probabilistic Machine Learning

Andrés R. Masegosa^{a,*}, Ana M. Martínez^{b,*}, Darío Ramos-López^{a,*}, Rafael Cabañas^{a,*},
Antonio Salmerón^a, Helge Langseth^c, Thomas D. Nielsen^b, Anders L. Madsen^{d,b}

^aUniversity of Almería, ES-04120 Almería, Spain

^bAalborg University, DK-9220 Aalborg, Denmark

^cNorwegian University of Science and Technology, NO-7491 Trondheim, Norway

^dHUGIN EXPERT A/S, DK-9000 Aalborg, Denmark

Abstract

The AMIDST Toolbox is an open source Java software for scalable probabilistic machine learning with a special focus on (massive) streaming data. The toolbox supports a flexible modelling language based on probabilistic graphical models with latent variables. AMIDST provides parallel and distributed implementations of scalable algorithms for doing probabilistic inference and Bayesian parameter learning in the specified models. These algorithms are based on a flexible variational message passing scheme, which supports discrete and continuous variables from a wide range of probability distributions.

Keywords: Probabilistic Graphical Models, Scalable algorithms, Variational methods, Latent variables

1. Introduction

AMIDST¹ is a toolbox for the analysis of large-scale data sets using probabilistic graphical models (PGMs). These are the so-called openbox models in the sense that PGMs can be easily interpreted by the users. PGMs consist of two parts: a qualitative component in the form of a graph encoding conditional independencies, and a quantitative component consisting of a collection of local probability distributions adhering to the independence properties specified in the graph. Collectively, the two components provide a compact representation of the joint probability distribution over the set of variables in the domain being modelled.

AMIDST implements parallel and distributed algorithms for Bayesian inference and learning in PGMs with latent (or unobserved) variables. The key point of this software is the use of variational methods [6] for making approximate inference. This makes

*These four authors are considered as first authors and contributed equally to this work.

Email addresses: andresmasegosa@ual.es (Andrés R. Masegosa), ana@cs.aau.dk (Ana M. Martínez), dramoslopez@ual.es (Darío Ramos-López), rcabanas@ual.es (Rafael Cabañas), antonio.salmeron@ual.es (Antonio Salmerón), helgel@idi.ntnu.no (Helge Langseth), tdn@cs.aau.dk (Thomas D. Nielsen), anders@hugin.dk (Anders L. Madsen)

¹For brevity, we will refer to the AMIDST Toolbox as either AMIDST or the toolbox.

14 AMIDST suitable for analysing streaming data because our models can efficiently be
 15 updated when new data is available. Numerous tools for graphical models have been
 16 published during the last three decades². However, the vast majority of them do not
 17 support scalable inference and learning algorithms. To the best of our knowledge,
 18 there is no existing software for mining data streams based on PGMs (including latent
 19 variable models); most existing tools focus on stationary data sets [8]. A qualitative and
 20 quantitative comparison with related tools can be found in the online documentation.

21 2. Background

22 2.1. Probabilistic graphical models

23 AMIDST supports the specification of *Bayesian networks (BNs)* [9, 2], which are
 24 widely used PGMs for reasoning under uncertainty. Formally, let $\mathbf{X} = \{X_1, \dots, X_N\}$
 25 denote the set of random variables defining our problem domain. BNs can be repre-
 26 sented by a directed acyclic graph (DAG). Each node, labelled X_i , is associated with a
 27 factor or conditional probability $p(X_i|pa(X_i))$, where $pa(X_i) \subset \mathbf{X} \setminus X_i$ represents the
 28 so-called *parent variables* of X_i , i.e., the variables corresponding to the parent nodes of
 29 X_i in the graph. A BN defines a joint distribution $p(\mathbf{X})$ in the following form:

$$p(\mathbf{X}) = \prod_{i=1}^N p(X_i|pa(X_i)). \quad (1)$$

30 For modelling problems where variables have continuous state spaces, the AMIDST
 31 Toolbox allows the specification of conditional linear Gaussian (CLG) Networks [4, 5].
 32 Furthermore, latent (i.e., hidden) variables are supported. These variables cannot be
 33 observed and are included in the model to capture correlation structure. The use of
 34 latent variables allows the representation of a large range of problems with complex
 35 probabilistic dependencies.

36 2.2. Scalable inference with variational methods

37 Inference (a.k.a. belief updating) in PGMs typically corresponds to calculating the
 38 posterior distribution $p(\mathbf{X}_I = \mathbf{x}_I | \mathbf{X}_E = \mathbf{x}_E)$, where $\mathbf{X}_E \subset \mathbf{X}$ is the set of observed
 39 variables and \mathbf{X}_I is the set of variables of interest with $\mathbf{X}_I \subseteq \mathbf{X} \setminus \mathbf{X}_E$.
 40

41 Variational inference is a deterministic approximate inference technique, where we
 42 seek to iteratively optimise a variational approximation to the posterior distribution of
 43 interest [1]. Let \mathcal{Q} be the set of possible approximations; then the variational approxi-
 44 mation to a posterior distribution $p(\mathbf{x}_I | \mathbf{X}_E = \mathbf{x}_E)$ is defined as

$$q_{\mathbf{x}_E}^*(\mathbf{x}_I) = \arg \min_{q \in \mathcal{Q}} D(q(\mathbf{x}_I) || p(\mathbf{x}_I | \mathbf{X}_E = \mathbf{x}_E)),$$

45 where $D(q||p)$ is the Kullback-Leibler divergence between q and p . In the AMIDST
 46 Toolbox, the variational inference scheme employs a so-called mean-field approxima-
 47 tion, which roughly assumes that the variables of interest are pairwise independent

²See this link for an updated list <http://www.cs.ubc.ca/~murphyk/Software/bnsoft.html>.

48 given the observed evidence; in turn this means that the posterior variational distri-
49 bution factorises over the variables involved, i.e., $q_{\mathbf{x}_E}^*(\mathbf{x}_I) = \prod_{i \in I} q_{\mathbf{x}_E}^*(x_i)$. Optimising
50 the variational approximation can be achieved by using either coordinate or gradient
51 ascent (or a stochastic approximation thereof), which guarantees convergence towards
52 a (local) optimum.
53

54 Learning the parameters θ of a BN from a training data set D can be reduced to
55 the inference task of computing $p(\theta|D)$. With this consideration, the AMIDST Tool-
56 box implements a general architecture for supporting the *variational message passing*
57 (VMP) algorithm [11] in PGMs. This algorithm can be easily scaled-up as messages
58 are independent. In particular, two versions are provided; a parallel version exploiting
59 multi-core architectures, powered by Java 8 Streams [7]; and a novel distributed version,
60 named d-VMP [6], for large-scale data processing on computing clusters running either
61 Apache Flink or Apache Spark.

62 3. Software framework

63 3.1. Functionalities

64 The key functionalities of the toolbox are summarised as follows:

- 65 • **Openbox models:** with the specification of PGMs, AMIDST’s approach to
66 machine learning is based on the use of openbox models that can be inspected
67 and which can incorporate prior information or knowledge about the domain, in
68 contrast to other approaches which cannot be interpreted by the users.
- 69 • **Efficient belief updating:** this toolbox implements, among others, approximate
70 Bayesian inference algorithms based on variational methods (see Section 2.2).
71 This allows for an efficient updating of the models which is suitable in cases
72 where the whole data cannot be stored in memory.
- 73 • **Multi-core and distributed learning:** AMIDST provides parallel and dis-
74 tributed implementations of variational algorithms [11] that can be run on multi-
75 core CPUs, using Java 8’s built-in functionalities, or in massive data sets by in-
76 terfacing with Apache Flink and Apache Spark. Further details and experimental
77 results about these methods can be found in [6, 7].

78 3.2. Architecture

79 AMIDST has been designed following a modular structure. This allows future ex-
80 tensions to be made independently of the core design, thereby leaving the kernel small
81 and robust. Another added value of the modularity is that it enables a more seamless
82 interaction with external software. Currently, AMIDST interfaces with MOA, Weka,
83 and HUGIN³. The toolbox is distributed using Maven⁴. The use of this technology
84 simplifies the installation making the interaction with external software transparent.

³ MOA: <http://moa.cms.waikato.ac.nz>, Weka: <http://www.cs.waikato.ac.nz/ml/weka/>, and HUGIN: <http://www.hugin.com>.

⁴<https://maven.apache.org>

85 4. Illustrative examples

86 In this section we illustrate the use of AMIDST in multi-core and parallel architec-
87 tures⁵. In particular, we consider the classification model proposed in [3] and a dataset
88 used in genetics [10] (which contains about 500,000 instances and which has been split
89 into files of 100,000 instances).
90

91 The `DataStream` class in package `eu.amidst.core.datastream` is an interface for
92 dealing with data streams in a single computer. The toolbox is designed to process the
93 data sequentially without having to load all observations into main memory simultane-
94 ously. The functionality for loading data is provided by class `DataStreamLoader`. The
95 following code provides an example of reading data from a `.arff` file (Weka file format):

```
96  
97  
98 1 DataStream data = DataStreamLoader.open("codrnaNorm_100k_1.arff");  
99
```

101 When we have a massive data set which does not fit into a single computer, we
102 can use a Big Data framework like Apache Flink to deal with data sets stored in
103 a distributed computing cluster. For reading these data sets we can use the class
104 `eu.amidst.flink.data.DataFlink`, as shown in the next code fragment:

```
105  
106  
107 1 //Set-up Flink Session  
108 2 ExecutionEnvironment env = ExecutionEnvironment.getExecutionEnvironment();  
109 3 // Load the distributed data  
110 4 DataFlink<DataInstance> data =  
111 5     DataFlinkLoader.open(env, "hdfs://codrnaNorm_100k_1.arff", false);  
112  
113
```

114 AMIDST contains a wide range of predefined models, most of them including latent
115 variables (and custom models can also be defined by the user). These models are
116 available in the *latent-variable-models* module. Learning is straightforward as shown in
117 the next code fragment, which also illustrates the toolbox’s seamless handling of massive
118 data sets during model learning/updating; the code is valid for both `DataStream` and
119 `DataFlink` objects. Lines 4 to 8 show how the model can be updated in case new data
120 sets become available.

```
121  
122  
123 1 Model model = new LatentClassificationModel(data.getAttributes())  
124 2     .setClassName("codrna_Y")  
125 3  
126 4 for(int i=1; i<=5; i++) {  
127 5     if (i > 1) data = DataStreamLoader.open("codrnaNorm_100k_"+i+".arff");  
128 6     model.updateModel(data);  
129 7     System.out.println(model.getModel());  
130 8 }  
131  
132
```

⁵Visit <https://github.com/amidst/example-project> for downloading an easy to run project with these examples.

133 AMIDST’s webpage (www.amidsttoolbox.com) contains a large class of code ex-
134 amples covering all the functionalities of the toolbox.

135 Acknowledgments

136 This work was performed as part of the AMIDST project. AMIDST has received
137 funding from the European Union’s Seventh Framework Programme for research, tech-
138 nological development and demonstration under grant agreement no 619209. AM, DRL
139 and AS thank the support from CDTIME. DRL thanks also to CEIMAR.

- 140 [1] H. Attias. A variational Bayesian framework for graphical models. *Advances in*
141 *neural information processing systems*, pages 209–215, 2000.
- 142 [2] F.V. Jensen and T.D. Nielsen. *Bayesian Networks and Decision Graphs*. Springer
143 Publishing Company, Incorporated, second edition, 2007.
- 144 [3] H. Langseth and T.D. Nielsen. Latent classification models. *Machine Learning*,
145 59(3):237–265, 2005.
- 146 [4] S.L. Lauritzen. Propagation of probabilities, means, and variances in mixed
147 graphical association models. *Journal of the American Statistical Association*,
148 87(420):1098–1108, 1992.
- 149 [5] S.L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.
- 150 [6] A.R. Masegosa, A. M. Martínez, H. Langseth, T.D. Nielsen, A. Salmerón,
151 D. Ramos-López, and A.L. Madsen. Scaling up Bayesian variational inference
152 using distributed computing clusters. *International Journal of Approximate Rea-*
153 *soning*, 88:435–451, 2017.
- 154 [7] A.R. Masegosa, A.M Martínez, and H. Borchani. Probabilistic graphical models
155 on multi-core CPUs using Java 8. *IEEE Computational Intelligence Magazine*,
156 11(2):41–54, 2016.
- 157 [8] K.P. Murphy. Software for graphical models: A review. *International Society for*
158 *Bayesian Analysis Bulletin*, 14(4):13–15, 2007.
- 159 [9] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible*
160 *Inference*. Morgan Kaufmann Publishers, San Mateo, CA., 1988.
- 161 [10] A.V. Uzilov, J.M. Keegan, and D.H. Mathews. Detection of non-coding RNAs
162 on the basis of predicted secondary structure formation free energy change. *BMC*
163 *bioinformatics*, 7(1):173, 2006.
- 164 [11] J.M. Winn and C.M. Bishop. Variational message passing. *Journal of Machine*
165 *Learning Research*, 6:661–694, 2005.

166 **Required Metadata**

167 **Current executable software version**

Nr.	(executable) Software metadata description	Please fill in this column
S1	Current software version	0.7.2
S2	Permanent link to executables of this version	https://github.com/amidst/toolbox/releases/tag/v0.7.2
S3	Legal Software License	Apache 2.0
S4	Computing platform/Operating System	Linux, OS X, Microsoft Windows
S5	Installation requirements & dependencies	Maven, Java 8
S6	If available, link to user manual - if formally published include a reference to the publication in the reference list	http://www.amidsttoolbox.com/documentation/
S7	Support email for questions	contact@amidsttoolbox.com

Table 1: Software metadata

168 **Current code version**

Nr.	Code metadata description	Please fill in this column
C1	Current code version	0.7.2
C2	Permanent link to code/repository used of this code version	https://github.com/amidst/toolbox
C3	Legal Code License	Apache 2.0
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	Java 8
C6	Compilation requirements, operating environments & dependencies	Maven
C7	If available Link to developer documentation/manual	http://www.amidsttoolbox.com/documentation/
C8	Support email for questions	contact@amidsttoolbox.com

Table 2: Code metadata