Data Stream Mining: A Review of Learning Methods and Frameworks

Svitlana Volkova
Center for Language and Speech Processing
Johns Hopkins University
svitlana@cs.jhu.edu
October 12, 2012

Abstract

The goal of the paper is to review methods, algorithms and frameworks for processing and analyzing real time data streams. We first give an overview of classic streaming algorithms and show how they can be applied for solving large-scale natural language processing (NLP) tasks. We next discuss differences between stream- and batch-mode learning and analyze two stream-based learning frameworks – Jerboa and MOA. Finally, we describe existing applications for learning from social media streams.

1 Introduction

Data stream mining is a relatively new research area that is, unfortunately, less studied compared to data mining and learning in a traditional batch mode. The majority of evaluations in the traditional batch mode use less than 1 million training examples. In contrast, stream-based learning models are designed to handle very large (potentially infinite) streams of examples. Moreover, demonstrating system performance on a small amount of data is not a convincing case for capacity while learning from recently emerged streaming applications including processing massive text streams [Bif, 2012].

A variety of natural processing tasks require making decisions in real time at a high speed on a continuous flow of text data. One of the reasons to this growing demand is the rapid growth of social media services – the most popular Twitter[1], Facebook[2] that allow mining useful up-to-date information from public threads in real time. For example, real-time opinion mining and sentiment analysis in social media streams is becoming increasingly popular [Pang & Lee, 2008]. The other NLP tasks that also require processing text streams in real time include named entity disambiguation [Sarmento et al., 2009] and statistical machine translation [Levenberg et al., 2011].

In this paper we review the exciting algorithms for processing text streams and give a comprehensive analysis of two stream-based learning frameworks. We structure the paper as follows: in Section 2 we review classic streaming algorithms and show the examples of applying these algorithms to solve large-scale NLP tasks; in Section 3 we talk about stream-based learning where we first enumerate differences between stream and batch learning, next we present two stream-based learning frameworks and, finally, we give the examples of stream-based learning from social medial streams.

[1] www.twitter.com
[2] www.facebook.com

2 Classic Streaming Algorithms for NLP Tasks

A data stream is a real-time, continuous, ordered sequence of items σ. It is impossible to control the order in which items arrive and to store a stream entirety as defined in [Golab & Özsu, 2003]. The goal of streaming algorithms is to compute some function of a massively long sequence of items (tokens) $\sigma =< a_1, a_2, a_3, \ldots, a_m >$ taking into account limited time and space constraints.

The foundations of streaming algorithms and a comprehensive overview of streaming systems and applications are given in [Muthukrishnan, 2005]. More general purpose streaming algorithms such as: finding frequent items deterministically (Mistra-Gries algorithm) and via sketching (count-min algorithm), estimating the number of distinct items using hashing (BJKST algorithm), and graph streaming algorithms discussed in details in [Chakrabarti, 2009] as a part of his streaming course at Dartmouth college. The other streaming courses that have excellent notes are taught by Indyk and MIT [Indyk, 2007] and McGregor at UMass Amherst [McGregor, 2011]. Finally, recent ACM publication in XRDS journal gives a concise and the most up-to-date overview of sketching and streaming algorithms for processing massive data and presents classic streaming algorithms such as: probabilistic counting (Moris' algorithm), frequent items (MJRTY algorithm), and linear sketches [Nelson, 2012].

Classic streaming algorithms are designed to process any kinds of data streams for a variety of applications including large-scale scientific experiments, search engines, online content delivery, and consumer tracking for large retails. Despite the fact that classic streaming algorithms are not designed learn and predict latent attributes learned from the data stream, they can be reused to reduce space and time constraints for large-scale NLP tasks, as shown in the examples below:

- Goyal et. al. applied classic streaming algorithm from [Manku & Motwani, 2002] that computes approximate frequency counts of frequently occurring n-grams to solve large-scale SMT task by constructing high-order approximate n-gram language models in a range of billions of words [Goyal et al., 2009].

- Goyal and Hal Daume III explored sketching techniques, specifically count-min sketch algorithm from [Cormode & Muthukrishnan, 2005], to solve three important NLP tasks: predicting word semantic orientations, distributional approaches to word similarity and unsupervised dependency parsing [Goyal & Daumé, 2011].

More examples of streaming and randomized algorithms, as well as stream-based learning approaches and frameworks designed to solve large-scale NLP tasks are discussed in Section 3.

3 Learning from Streaming Text Data

3.1 Stream vs. Batch Learning

State-of-the-art machine learning algorithms for clustering and classification use batch procedures and are not designed to make a prediction in streaming mode due to the following assumptions [Bishop, 2006]:

a) examples are independent and identically distributed (*i.i.d.*) random variables;
b) examples are randomly generated from stationary probability distribution D.

Real-time decision models are dynamic models that use unbounded training sets where training examples are not *i.i.d.*. To make decisions in real time, learning algorithms for must have the following capabilities [Gama et al., 2009]:

a) deal with data whose distribution changes over time;
b) track changes and update the decision accordingly;
c) incorporate new data at the speed it arrives;
d) use limited resources – time and memory.

3.2 Stream-based Learning Frameworks

In this section we analyze two frameworks[3] for stream-based clustering and classification from massive data.

1. **Jerboa** is a toolkit of randomized and streaming algorithms for large-scale language mining tasks developed by Benjamin Van Durme [Van Durme, 2012a]. Jerboa toolkit uses a variety of efficient filtering, sampling and counting techniques, well-known in the database community, for processing streaming data including Bloom filters [Bloom, 1970] and reservoir sampling [Vitter, 1985] to process data fast and efficiently, Morris-style [Morris, 1978], Talbot Osborne Morris Bloom (TOMB) and reservoir counters [Van Durme & Lall, 2011] to keep track of very large keysets of ngram types and Local Sensitive Hashing (LSH) methods [Van Durme & Lall, 2010] to reduce the memory use.

   Jerboa toolkit was effectively used for streaming analysis of discourse participants [Van Durme, 2012b]. A standard bag-of-word (BOW) classification model for predicting user gender based on user utterance is decomposed into a series of streaming updates; reservoir counting algorithm [Van Durme & Lall, 2011] is used as an approximation method for compressing the streaming memory requirements of the classifier by 75% with no significant cost in accuracy.

2. **MOA** (Massive Online Analysis) is an open-source software environment for implementing algorithms and running experiments for online learning from massive, potentially infinite, evolving data streams [Bifet et al., 2010].

   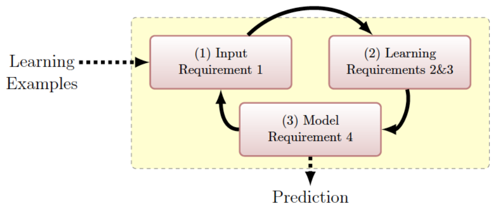

   Figure shows the typical use of stream classification algorithm: (1) reading the next available example from the stream (req.1); (2) processing the examples and updating the data structures withing the memory (req.2) and time (req.3) bounds; (3) ready to accept the next example and to predict the class of unseen examples (req.4).

[3] To the best of our knowledge, these are the only publicly available stream-based learning frameworks that contain state-of-the-art learning algorithms for streaming mode prediction.

MOA permits evaluation of data stream learning algorithms on large streams in the order of 10-100 millions examples. MOA allows joining and filtering streams and contains several stream classification methods: Naive Bayes, Decision Stump, Hoeffding Tree, Bagging, Boosting and clustering methods: StreamKM++, CluStream, ClusTree, Den-stream, D-stream and CobWeb [Bifet et al., 2010].

MOA was effectively used for detecting sentiment change in Twitter streaming data using frequent item mining approach [Bifet et al., 2011] and other stream mining methods including Multinomial Naive Bayes, Stochastic Gradient Descent and Hoeffding Tree [Bifet & Frank, 2010].

Finally, according to Bifet [Bif, 2012] there are other ways to speedup the mining of streaming learners is to distribute the training process, for example using Hadoop MapReduce[4] programming model to process data in parallel on large clusters. Arache S4 [Neumeyer et al., 2010] platform, inspired by MapReduce model, is designed to process data streams.

3.3 Learning from Social Media Streams

Many application for analyzing massive data sizes of user generated content (UGC) produced daily by microblogging and social networking serviced exist. In this section we review several examples of recently emerged online services for social media stream mining.

- **Cloud4Trends** is a system for analyzing social network trends and dynamics in real-time fashion using clustering approach [Vakali et al., 2012]. Trend detection application has 3-tier design: (1) data collection in streaming fashion; (2) application of an online clustering technique on the data to detect recent trends; and (3) refinement and ranking of clusters for data visualization. Cloud4Trends runs over the VENUS-C infrustructure [5].

- **SPOONS** (Swift Perceptions of Online Negative Situations) is a Netflix service outtage detection system via real-time stream analysis [Augustine et al., 2012]. The system does not perform multiclass classification in a classic streaming fashion, instead it incrementally retrains a variety WEKA learners including Naive Bayes, Bayes Net, J48, KNN [Hall et al., 2009] on a new batch of training data to predict Netflix service outtage.

Due to space limitations, we can not describe all streaming systems that process social media. However we can enumerate the most popular services including Google Hot Trends[6], IceRocket [7], BingSocial [8], SocialMention [9], Topsy [10].

[4] http://hadoop.apache.org/
[5] http://www.venus-c.eu
[6] http://www.google.com/trends/hottrends
[7] http://www.icerocket.com/popular/
[8] http://www.bing.com/social
[9] http://socialmention.com/trends/
[10] http://topsy.com/

4 Summary

In this paper we reviewed learning methods and tools for massive, infinite data stream mining. We presented classic streaming algorithms and demonstrated how they can be reused in order to reduce space and processing time for large scale NLP problems. We analyzed two stream-based learning frameworks – Jerboa and MOA, and described sampling, counting and filtering techniques used to reduce memory size and increase the efficiency of computations while learning from data streams in real time. We also demonstrated a typical use of stream classification algorithm in MOA and presented a set of classification and clustering algorithms designed to make a prediction in streaming fashion on millions of examples. The major property of stream-based learning algorithms is processing the data that arrives at high speed under very strict constraints of space and time. Finally, we described the existing applications for analyzing massive data from social media streams – Cloud4Trends and SPOONS.

References

[Bif, 2012] (2012). *Mining Big Data in real Time*, New York, NY, USA. ACM.

[Augustine et al., 2012] Augustine, E., Cushing, C., Dekhtyar, A., McEntee, K., Paterson, K., & Tognetti, M. (2012). Outage detection via real-time social stream analysis: leveraging the power of online complaints. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion (pp. 13–22). New York, NY, USA: ACM.

[Bifet & Frank, 2010] Bifet, A. & Frank, E. (2010). Sentiment knowledge discovery in twitter streaming data. In *Proceedings of the 13th international conference on Discovery science*, DS'10 (pp. 1–15). Berlin, Heidelberg: Springer-Verlag.

[Bifet et al., 2011] Bifet, A., Holmes, G., Pfahringer, B., & Gavaldà, R. (2011). Detecting sentiment change in twitter streaming data. *Journal of Machine Learning Research - Proceedings Track*, 17, 5–11.

[Bifet et al., 2010] Bifet, A., Holmes, G., Pfahringer, B., Kranen, P., Kremer, H., Jansen, T., & Seidl, T. (2010). Moa: Massive online analysis, a framework for stream classification and clustering. *Journal of Machine Learning Research - Proceedings Track*, 11, 44–50.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

[Bloom, 1970] Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7), 422–426.

[Chakrabarti, 2009] Chakrabarti, A. (2009). Data Stream Algorithms. Lecture Notes. http://www.cs.dartmouth.edu/~ac/Teach/CS49-Fall11/.

[Cormode & Muthukrishnan, 2005] Cormode, G. & Muthukrishnan, S. (2005). An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1), 58–75.

[Gama et al., 2009] Gama, J. a., Sebastião, R., & Rodrigues, P. P. (2009). Issues in evaluation of stream learning algorithms. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09 (pp. 329–338). New York, NY, USA: ACM.

[Golab & Özsu, 2003] Golab, L. & Özsu, M. T. (2003). Processing sliding window multi-joins in continuous queries over data streams. In *VLDB* (pp. 500–511).

[Goyal & Daumé, 2011] Goyal, A. & Daumé, III, H. (2011). Approximate scalable bounded space sketch for large data nlp. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11 (pp. 250–261). Stroudsburg, PA, USA: Association for Computational Linguistics.

[Goyal et al., 2009] Goyal, A., Daumé, III, H., & Venkatasubramanian, S. (2009). Streaming for large scale nlp: language modeling. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09 (pp. 512–520). Stroudsburg, PA, USA: Association for Computational Linguistics.

[Hall et al., 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1), 10–18.

[Indyk, 2007] Indyk, P. (2007). Sketching, Streaming and Sub-linear Space algorithms. Lecture Notes. http://stellar.mit.edu/S/course/6/fa07/6.895/materials.html.

[Levenberg et al., 2011] Levenberg, A., Osborne, M., & Matthews, D. (2011). Multiple-stream language models for statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11 (pp. 177–186). Stroudsburg, PA, USA: Association for Computational Linguistics.

[Manku & Motwani, 2002] Manku, G. S. & Motwani, R. (2002). Approximate frequency counts over data streams. In *Proceedings of the 28th international conference on Very Large Data Bases*, VLDB '02 (pp. 346–357).: VLDB Endowment.

[McGregor, 2011] McGregor, A. (2011). More Advanced Algorithms. Lecture Notes. http://people.cs.umass.edu/~mcgregor/courses/CS711S12/.

[Morris, 1978] Morris, R. (1978). Counting large numbers of events in small registers. *Commun. ACM*, 21(10), 840–842.

[Muthukrishnan, 2005] Muthukrishnan, S. (2005). Data streams: algorithms and applications. *Found. Trends Theor. Comput. Sci.*, 1(2), 117–236.

[Nelson, 2012] Nelson, J. (2012). Sketching and streaming algorithms for processing massive data. *XRDS*, 19(1), 14–19.

[Neumeyer et al., 2010] Neumeyer, L., Robbins, B., Nair, A., & Kesari, A. (2010). S4: Distributed stream computing platform. In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on* (pp. 170 –177).

[Pang & Lee, 2008] Pang, B. & Lee, L. (2008). Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2), 1–135.

[Sarmento et al., 2009] Sarmento, L., Kehlenbeck, A., Oliveira, E., & Ungar, L. (2009). An approach to web-scale named-entity disambiguation. In *Proceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM '09 (pp. 689–703). Berlin, Heidelberg: Springer-Verlag.

[Vakali et al., 2012] Vakali, A., Giatsoglou, M., & Antaris, S. (2012). Social networking trends and dynamics detection via a cloud-based framework design. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion (pp. 1213–1220). New York, NY, USA: ACM.

[Van Durme, 2012a] Van Durme, B. (2012a). *Jerboa: AToolkit for Randomized and Streaming Algorithms*. Technical report, Human Language Technology Center of Excellence.

[Van Durme, 2012b] Van Durme, B. (2012b). Streaming analysis of discourse participants. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (pp. 48–58). Jeju Island, Korea: Association for Computational Linguistics.

[Van Durme & Lall, 2010] Van Durme, B. & Lall, A. (2010). Online generation of locality sensitive hash signatures. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10 (pp. 231–235). Stroudsburg, PA, USA: Association for Computational Linguistics.

[Van Durme & Lall, 2011] Van Durme, B. & Lall, A. (2011). Efficient online locality sensitive hashing via reservoir counting. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11 (pp. 18–23). Stroudsburg, PA, USA: Association for Computational Linguistics.

[Vitter, 1985] Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1), 37–57.