Scalability and Cost of a Cloud-based Approach to Medical NLP

Kyle Chard, Michael Russell
Computation Institute
University of Chicago and Argonne National Laboratory
Chicago, IL
{kyle, mike}@ci.uchicago.edu

Yves A. Lussier
Section of Genetic Medicine, Dep. of Medicine
University of Chicago
Chicago, IL
ylussier@bsd.uchicago.edu

Eneida A Mendonça
Dep. of Biostatistics and Medical Informatics
University of Wisconsin-Madison
Madison, WI
emendonca@biostat.wisc.edu

Jonathan C. Silverstein
NorthShore University HealthSystem
Evanston, IL
jsilverstein@northshore.org

Abstract

Natural Language Processing (NLP) in the medical field has the potential to dramatically influence the way in which everyday clinical care and medical research is conducted. NLP systems provide access to structured content embedded in raw medical texts, therefore enabling automated processing. There are however, several barriers prohibiting wide spread adoption of NLP technology primarily driven by the complexity and cost. This paper describes an approach and implementation which leverages cloud-based deployment and service-based interfaces to extract, process, synthesize, mine, compare/contrast, explore, and manage medical text data in a flexibly secure and scalable architecture. Through a virtual appliance architecture users are able to discover, deploy and utilize NLP engines on demand without requiring knowledge of the underlying, potentially complex, NLP engine. As highlighted in this paper, the system architecture can scale in several configurations: by increasing the number of instances deployed, the number of NLP engines, and the number of databases.

1. Introduction

Natural Language Processing (NLP) provides access to structured semantic data embedded in raw text documents. This structured content is a crucial first step towards automated processing of medical documents in clinical decision support and research. However, medical NLP tools are often complex and computationally intensive due to the multiple processing algorithms applied to improve mapping quality. This computational burden makes real-time use infeasible [3] and requires large scale resources to scale efficiently.

With the advent of Cloud computing, users have on demand access to large scale computing resources with minimal infrastructural investment. The creation of self contained machine images, or *virtual appliances*, allows simple discovery and use without requiring individual machine installation. The Cloud computing model is particularly attractive because of the utility pricing model and the elastic nature of the resources. That is, additional resources can be added on demand and users pay only for the resources used. Essentially this provides a simple way to parallelize applications, increasing scalability, and increasing overall system performance. However, at present most NLP engines are designed for use in a centralized deployment.

Smntx [5] is our distributed service based architecture designed to improve accessibility, scalability and flexibility of medical NLP applications. Rather than directly providing NLP capabilities, Smntx leverages existing NLP engines and coordinates distributed access to these tools. Smntx stores and indexes coded results such that data mining and analysis can be performed in real time. Smntx exposes a lightweight Representational State Transfer (REST) interface that supports standardized invocation and usage in a wide variety of environments, such as Grids and Clouds, without requiring heavyweight client APIs.

Using Smntx in a cloud-based approach to medical NLP with virtual appliances and REST services enables a flexible scalable architecture that is agnostic to the NLP engine used. This architecture facilitates flexible deployment scenarios and simplified creation of non-expert tools that abstract the complexities of NLP technology and reduce barriers to entry. It is hoped that this will lead to widespread adoption of NLP technology in many domains.

This paper outlines the Smntx architecture designed to improve NLP usability and scalability, and demonstrates the relative costs and performance improvements gained through Cloud-based deployments.

2 Background and Related Work

Cloud computing has drawn much attention as a means of scaling applications on demand. In the broader bioinformatics domain, Cloud computing has been used in comparative genomics [12], biomedical data and application sharing [10], and translational bioinformatics [8]. In each of these studies, the advantages of elastic computing was shown to provide a scalable environment with reduced investment in dedicated resources. Carrell [4] proposed a virtual appliance model similar to that of Smntx, however the preliminary work presented only included a general architecture and a discussion of the security implications when processing clinical narratives in the Cloud.

There are many examples of mature medical NLP engines, including MetaMap [2], MedLEE (Language Extraction and Encoding) [9] and cTakes (Mayo clinical Text Analysis and Knowledge Extraction System) [11]. These NLP tools are based on raw processing, in which expert users format (pre-process) and submit unstructured medical records through a command line interface. They then interpret the output results with the aid of proprietary post processing scripts. This is a time consuming and error prone task that requires a great deal of domain and NLP knowledge. In addition the high processing costs represent a severe barrier to entry for new users. Consequently there is a dearth of applications that make direct use of NLP data.

Most NLP engines are designed for use in a centralized deployment and do not offer an integrated service interface (MetaMap has a Java API). One project that has taken a service oriented approach to medical NLP is the Cancer Text Information Extraction System (caTIES) [6]. caTIES is designed to support collaborative tissue banking and text mining. The general NLP workflow pares free text, maps phrases to a limited set of concepts and extracts a result hierarchy to XML. The caTIES workflow is constructed using a combination of GATE [7](General Architecture for Text Engineering) and custom components. MetaMap Transfer is used to map concepts to fragments of free text. The architecture is composed of a group of Grid services wrapping both functionality and data sets, but it does not support parallel or Cloud based deployment

3 Smntx

Smntx takes a user-oriented approach to medical NLP by focusing on usability, flexibility, accessibility and scalability. Smntx is a distributed service-based medical NLP architecture that abstracts technical NLP detail through a standardized REST interface and facilitates semi-transparent parallel invocation of different NLP engines.

All NLP analysis results are stored and indexed to provide real-time mining through complex queries and faceted search. Smntx includes a fully functional AJAX based web interface through which non-expert users can analyze, mine, explore and synthesize data over multiple documents without requiring in depth knowledge of the underlying NLP engine. The web interface is designed to expose the core functionality in an intuitive manner, supporting both individual file analysis with customizable text highlighting and a multi-document data mining. Finally, the Smntx model supports user enrichment (annotation), allowing users to improve mapping quality and store results for subsequent processing.

The high level architecture of Smntx is shown in Figure 1 The architecture is composed of three major components, these are: the user interface, NLP service wrapper, and distributed data repositories. Figure 1 also illustrates the versatility of the service-based architecture as three diverse end user tools are shown to be (simultaneously) interacting with the same backend services. In this figure a clinician is shown using the web interface, a researcher is using a high level programming language, and an entire institution is utilizing automated scientific workflows to process bulk data.

Smntx relies on existing NLP engines to parse unstructured medical text and map terms against a medical metathesaurus. Currently, Smntx includes a MetaMap NLP wrapper. MetaMap was chosen as the first wrapper because it is a generic NLP platform developed by the National Library of Medicine and it combines various processes and algorithms to map concepts from the Unified Medical Language System (UMLS) Metathesaurus to raw medical text.

Smntx supports the use of multiple (parallel) NLP engines to both, increase mapping quality, and also improve performance. All NLP requests are queued when received and allocated to registered NLP engines according to predefined scheduling algorithms. Before starting Smntx, available NLP engines must be registered in a properties

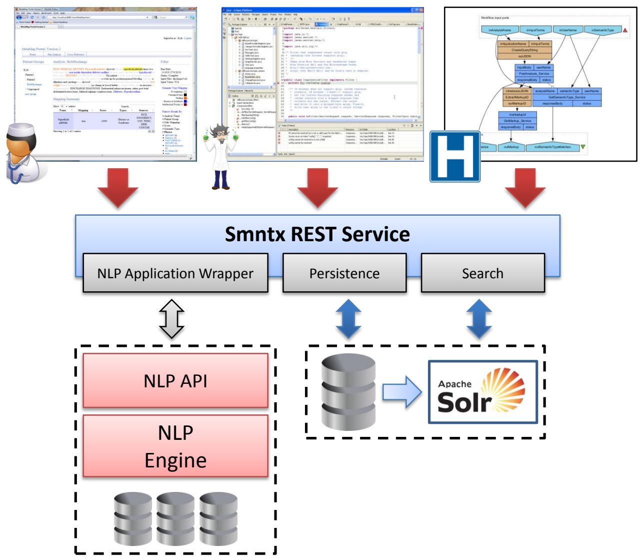

Figure 1. Smntx high level architecture.

file. Upon completion of analysis Smntx parses the results returned from the NLP engines to create HTML markup of the raw text and to store provenance data and results.

To reduce the computational cost of processing documents, Smntx attempts to reduce repeated processing through a single non-filtered NLP stage. After processing, real-time mining can be conducted using the indexed results rather than re-executing the NLP stage. To provide data persistence coded results and provenance data are stored in a durable relational database. All results and raw text are also indexed to optimize performance and support complex queries, full text search, and faceted search. In the current implementation, the relational database used is Apache Derby, a lightweight java database capable of operating in both embedded and network mode. Indexing and full text search is provided by Apache Solr; an enterprise search platform based on the Apache Lucene search library. Both the relational database and Solr are highly scalable due to their distributed architectures. Solr also supports both distributed search and index replication.

The flexibility of the Smntx architecture is demonstrated through the number of potential deployment environments. In [5] several deployment scenarios are presented, including CaBIG, i2b2 Hive and NHIN Direct. In each case the service-based architecture can be easily consumed in the target environment therefore exposing Smntx functionality to a diverse user community.

Given the strict regulations present in the medical domain, Smntx has been designed with privacy and security in mind. At the lowest level, for identified patient information, it is assumed that both the NLP engine and the Smntx services are hosted in a HIPAA compliant environment. In a less secure environment only de-identified data should be used. All interactions with Smntx must be performed with an authenticated user and users must register before being able to use any of the services. Currently, data is only accessible to the user that created it but we are investigating data sharing approaches. In addition to Smntx authentication, the UMLS data store requires that all users have UMLS Terminology Services (UTS) licenses. To support this, Smntx is an authorized content distributor which allows end users to authenticate with Smntx using their UTS credentials. Smntx is the first user of the UTS REST authentication service provided by National Library of Medicine. Thus this third-party UMLS license verification provided via REST interface ensures underlying libraries are licensed to the user with which we interact.

4 Cloud deployment

Cloud computing is a scalable computing model in which virtualized resources are provisioned on demand by consumers. Software as a Service (SaaS) forms the top layer of the Cloud stack, offering specialized well defined application data and software to consumers while abstracting underlying implementation details. Users access the service using a thin client and are typically unaware of the physical locale of the hosted service.

Smntx follows a SaaS approach through a modular service based architecture. One of the major advantages of this approach is that the Smntx application can scale on demand as additional virtual machines are created and deployed. Moreover, the backend services can be optimized through transparent parallelization (NLP engine) and "sharding" (distributing) the data repositories and indexes. In addition, small scale instances of Smntx can be trivially instantiated, deployed and destroyed to rapidly process data sets for a fixed period of time.

The Smntx SaaS architecture has been deployed to both a public commercial Cloud - Amazon Elastic Compute Cloud (EC2) and a private HIPAA compliant Cloud at the University of Chicago in the Computation Institute's Initiative in Biomedical Informatics

5 Evaluation

There are three distinct stages in the Smntx model: (1) NLP processing, (2) Smntx request processing and result parsing, and (3) result storage and indexing. This section explores the performance benefit and cost of distributing the first two stages of NLP processing in a Cloud. These two scenarios are depicted in Figure 2.

The most obvious means of distributing this processing is using a single Smntx instance and distributing the NLP

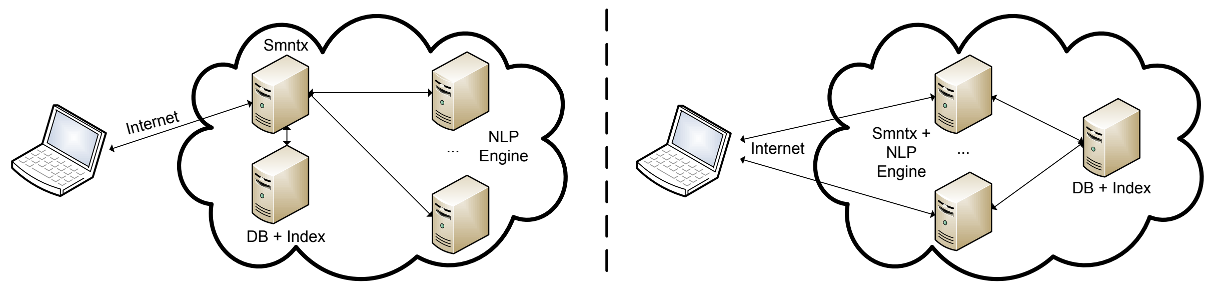

Figure 2. Distribution experiment configuration. The left side shows a single Smntx instance with multiple distributed NLP engines. The right side shows multiple distributed Smntx and NLP engines. Both scenarios share a single data repository with a database and index.

application. In this case the single Smntx instance accepts all requests and acts as a load distributor forking requests to a pool of NLP engines using a multi threaded model. This model provides a single point of contact for all consumers, however, it also represents a single point of failure and a potential bottleneck due the number of requests arriving and the non trivial processing (indexing and markup generation) performed by the Smtnx application.

A second approach can be applied to distribute both the Smntx instance and the NLP engine. This has the effect of parallelizing the cost of requests, processing and result parsing. However, in both cases a shared data repository must be used to ensure that all Smntx instances maintain consistent state. A third model (not explored here) would also distribute the data repositories to remove any potential bottleneck.

The experiment presented in Section 5.2 directly compares the performance of the two models depicted in Figure 2 (distributing MetaMap, and, distributing both Smntx and MetaMap). Section 5.3 studies the performance vs cost of deployment to common EC2 instance sizes.

5.1 Experimental Setup

The following experiments use Amazon EC2 as the Cloud provider. EC2 was chosen because it is the most mature commercial cloud available. The distribution experiments presented in Section 5.2 use *small* instances, each instance has 1.7 GB memory, 1 EC2 Compute Unit, 160 GB instance storage, 32-bit platform running Ubuntu 10.4. All instances are run in the US East region.

The experimental instances are managed using the EC2 web interface and Python based scripts that start and stop the required number of instances. Each instance is pre-configured such that it can be deployed in minutes and instantly used. A custom security policy has been defined to expose the ports required by each of the tools in the Smntx architecture.

To replicate a realistic usage scenario we created a client application and hosted it on a machine outside the Cloud infrastructure. The client uses a lightweight HTTP client to submit requests. Each request represents processing of an individual medical document. The multi threaded client loops through 1000 requests asynchronously to avoid blocking on a single request. In the second deployment scenario the client application follows a simple round robin scheduling algorithm, in essence simulating a load balancer. In a production deployment a dedicated load balancer such as Amazon Elastic Load Balancer (ELB) would be used.

To create reproducible and comparable results each experiment uses the same document processed 1000 times. The document used is a moderately sized 3KB discharge summary, which when processed, creates over 1000 mappings. All results presented are calculated based on the time to process all 1000 documents (over 1 million total mappings). Each experiment has been run multiple times to obtain an average result.

Each of the core services (Derby, Solr, Jetty, MetaMap) are started with 256 MB of memory and limited to no more than 512 MB.

5.2 Smntx Distribution

To measure the performance of distribution we deployed a pre-configured self contained Smntx VM on an increasing number of hosts. Figure 3 shows the total time taken to analyze 1000 discharge summaries under two different distribution scenarios: (1) distributing only MetaMap and (2) distributing both Smntx and MetaMap. The graph shows the total computation time decreases with the increase in the number of hosts used. The optimal (no overhead) function is calculated based on the average processing time for a single document (˜21s) assuming no overhead or scalability constraints.

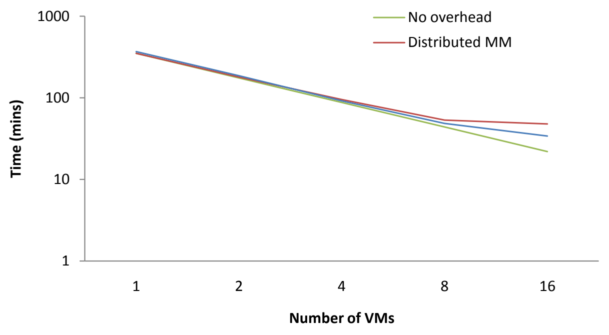

Figure 3. Time to process 1000 medical documents with an increasing number of parallel VMs.

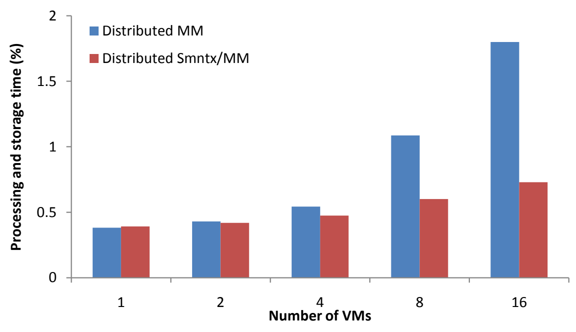

Figure 4. Processing and storage time as a percentage of total processing time for an increasing number of parallel VMs.

The difference between distributing MetaMap and distributing both MetaMap and Smntx only becomes apparent as the number of hosts increases. For a small number of hosts, increasing parallelization decreases the total time at nearly the optimal rate. As the number of hosts increases, the advantage of parallelization starts to decrease due to the additional burden related to the number of active instances. This is particularly the case when distributing only MetaMap. The reason for this difference is the substantial overhead of handling 16 simultaneous MetaMap connections, processing the responses and storing/indexing the results. When distributing both Smntx and MetaMap this difference is reduced because the overhead of distribution is also shared. However there is still some overhead due to the bottleneck formed at the data repository because significant data sets are pushed simultaneously. This relationship is highlighted when examining the average processing and storage time as a percentage of total time shown in Figure 4. The average processing time is much greater when a single Smntx instance is used due to the competition between analyses.

5.3 Instance Size

In a parallel environment, the pay-per-use utility model used by commercial Cloud providers is particularly advantageous, especially if the overhead for parallelization is small. For example, if we assume there is no overhead, the cost of running 1 VM for 5 hours and running 5 VMs for 1 hour each, is identical. The results presented in Section 5.1 showed that for a reasonable number of hosts the parallelization overhead of the Smntx architecture is small.

Amazon provides a number of different image sizes designed to suit different usage scenarios. Larger instances provide more computational power, however the cost is not proportional to the size of the instance. To evaluate the most cost effective image size we configured a 64 bit EBS image and a 64 bit image both running Ubuntu 10.4 to be hosted on the micro, large and extra large images available on EC2. The 32 bit image used in the previous section was also used to compare the small instance size. To determine the cost per document, the same 3KB medical document was processed 1000 times and the cost per document calculated. The results are presented in Table 1. All prices are based on the US East region.

	Micro	Small	Large	XLarge
Cores	Up to 2	1	4	8
Memory (GB)	0.613	1.7	7.5	15
Architecture	64	32	64	64
Storage (GB)	EBS	160	850	1690
Instance Cost ($/hr)	0.02	0.085	0.34	0.68
Documents per hour	66.8	170.0	270.4	391.1
Price per document ($)	0.0003*	0.0005	0.0012	0.0017

Table 1. Cost of running Smntx on different EC2 images. *EBS backed storage is an additional cost ($0.10 per GB/Month)

These results show that parallelizing smaller instances is generally more cost effective especially given the ease by which Smntx can be parallelized. Using the micro and small instance costs less than 5 hundredths of a cent to process a single document. If MetaMap were able to exploit multiple cores the large instance sizes may result in better performance due to the decreased latency.

5.4 Summary

The Amazon interface and APIs simplify administration and abstract low level details. By publishing a self contained Smntx image as an appliance, users are able to select and start up multiple instances in a matter of minutes. As soon as instances have been started they are immediately available for use without requiring any configuration. The flexibility of Smntx allows users to add and remove processing nodes on demand to facilitate peak processing requirements. Assuming the data repositories are persisted, users can mine indexed results in real time using only a single Smntx instance irrespective of other deleted instances.

The cost for analyzing a single 3KB document was shown to be approximately $0.0005 (or 50 cents per 1000 documents). Storing data in Amazon Simple Storage Service (S3) is also cost effective, starting at 14 cents per GB for a month. Therefore results can be persisted without requiring large investment.

One of the major hurdles prohibiting public cloud-based medical NLP is ensuring privacy and security [1]. The individual appliance model used by Smntx may reduce security risks because access to the tool is controlled by the owner. Each running instance can also be configured with different access policies and port mappings to limit external access. Data encryption can also be used to improve data security. However, we acknowledge, that without using a private cloud, further investigation is needed of remaining security barriers.

6 Conclusion

Medical NLP has shown great promise as a means of extracting structured text from raw medical documents. At present, however adoption is limited in part due to the complexity and computational requirements of existing NLP engines. We believe that a cloud-based approach using virtual machines and REST services can create a scalable architecture that is agnostic to the NLP engine used while also supporting flexible deployment scenarios (single VM or parallel VM Cloud deployments). Using pre-configured virtual appliances such as the EC2 images described in this paper, users can trivially discover, deploy and use a complete complex NLP environment, including NLP processing, persistent result and provenance data storage, free text search and intuitive data mining in a matter of minutes and costing only a few cents. It is our hope that this simplified model will facilitate use by non-expert users and lead to greater adoption in real world scenarios.

There are many areas of future work to realize the full vision of hosting medical NLP applications in the Cloud. In the immediate future we aim to add additional NLP engine wrappers to the Smntx model, thereby providing improved mapping quality and additional user selection. We could also leverage capabilities provided by Amazon such as the Elastic Load Balancer to evenly distribute load over a pool of Smntx instances and the Relational Database Service to provide a scalable distributed data store.

7 Acknowledgments

This project is supported in part by the NIH Clinical and Translational Science Awards (5UL1RR024999, 5U54CA121852)

References

[1] Amazon.com. Creating hipaa-compliant medical data applications with amazon web services, 2009.

[2] A. Aronson. A. effective mapping of biomedical text to the umls metathesaurus: the metamap program. In *Proceedings of the AIMA Symposium*, 2001.

[3] A. Aronson and F. Lang. An overview of metamap: historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17:229–236, 2010.

[4] D. Carrell. A strategy for deploying secure cloud-based natural language processing systems for applied research involving clinical text. In *Proccedings of the 44th Hawaii International Conference on System Sciences (HICSS)*, 2011.

[5] K. Chard, M. Russell, E. Mendonça, Y. A. Lussier, and J. Silverstein. A cloud-based approach to medical nlp. In *Submitted to: AIMIA*, 2011.

[6] R. S. Crowley, M. Castine, K. Mitchell, G. Chavan, T. McSherry, and M. Feldman. Caties: a grid based system for coding and retrieval of surgical pathology reports and tissue specimens in support of translational research. *Journal of the American Medical Informatics Association*, 17(3):253–264, 2010.

[7] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. Gate: an architecture for development of robust hlt applications. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002.

[8] J. T. Dudley, Y. Pouliot, R. Chen, A. A. Morgan, and A. J. Butte. Translational bioinformatics in the cloud:an affordable alternative. *Genome Medicine*, 2(8):51–57, 2010.

[9] C. Friedman, G. Hripcsaka, W. DuMouchela, Johnsona, and Claytona. Natural language processing in an operational clinical information system. *Journal of Natural Language Engineering*, 1(1):83–108, 1995.

[10] A. Rosenthal, P. Mork, M. Li, J. Stanford, D. Koester, and P. Reynolds. Cloud computing: A new business paradigm for biomedical information sharing. *Journal of Biomedical Informatics*, 43:342–353, 2010.

[11] G. Savova, J. Masanz, P. Ogren, J. Zheng, S.Sohn, K. Kipper-Schuler, and C.G.Chute. Mayo clinical text analysis and knowledge extraction. *Journal of the American Medical Informatics Association*, 17(5):507–513, 2010.

[12] D. P. Wall, P. Kudtarkar, V. A. Fusaro, R. Pivovarov, P. Patil, and P. J. Tonellato. Cloud computing for comparative genomics. *BMC Bioinformatics*, 11(1):259, 2010.