WASA: A Web Application for Sequence Annotation

Fahad AlGhamdi, and Mona Diab
Department of Computer Science
The George Washington University
Washington, DC
{fghamdi, mtdiab}@gwu.edu

Abstract

Data annotation is an important and necessary task for all NLP applications. Designing and implementing a web-based application that enables many annotators to annotate and enter their input into one central database is not a trivial task. These kinds of web-based applications require a consistent and robust backup for the underlying database and support to enhance the efficiency and speed of the annotation. Also, they need to ensure that the annotations are stored with a minimal amount of redundancy in order to take advantage of the available resources(e.g, storage space). In this paper, we introduce WASA, a web-based annotation system for managing large-scale multilingual Code Switching (CS) data annotation. Although WASA has the ability to perform the annotation for any token sequence with arbitrary tag sets, we will focus on how WASA is used for CS annotation. The system supports concurrent annotation, handles multiple encodings, allows for several levels of management control, and enables quality control measures while seamlessly reporting annotation statistics from various perspectives and at different levels of granularity. Moreover, the system is integrated with a robust language specific date prepossessing tool to enhance the speed and efficiency of the annotation. We describe the annotation and the administration interfaces as well as the backend engine.

Keywords: : Code Switching, Annotation, Web Application, Sociolinguistics

1. Introduction

Code Switching (CS) is a phenomenon that occurs when multilingual speakers alternate between more than one language or dialect. This phenomenon can be observed in different linguistic levels of representation for different language pairs: phonological, morphological, lexical, syntactic, semantic, and discourse/pragmatics. CS presents serious challenges for language technologies, including parsing, Machine Translation (MT), Information Retrieval (IR) and others. A major barrier to research on CS has been the lack of large multilingual, multi-genre CS-annotated corpora. Creating such corpora involves managing many annotators working on multiple tasks at different times, consistent and robust backups of the underlying database, quality control, etc. In this paper, we present our effort in building an annotation system, WASA, that can manage and facilitate large-scale CS data annotation. WASA differs from other annotation systems in several respects. Our system has an option that can provide initial automatic tagging for specific tokens such as Latin words, URL, punctuation, digits, diacritics, emoticons, and speech effect tokens. This option increases the quality and the speed of annotation substantially. Moreover, the system is integrated with language-specific date preprocessing tool Smart Preprocessing (Quasi) Language Independent Tool (SPLIT) (Al-Badrashiny et al., 2016) to streamline raw data cleaning and preparation.

The remainder of this paper is organized as follows: Section 2 provides an overview of related work. Section 3 describes the System Architecture. Types of users including permissions and users tasks are introduced in Section 4. The data preprocessing and cleaning are discussed in Section 5. We provide an overview of the database design in Section 6. Inter-annotator agreement, current status, and our conclusion and future work are discussed in sections 7,8 and 9, respectively.

2. Related Works

Although, many annotation tools, such as (Aziz et al., 2012), (Cunningham et al., 2009), (Kahan et al., 2002), MnM (Vargas-Vera et al., 2002), GATE ((Cunningham et al., 2009); (Aswani and Gaizauskas, 2009), and (Dickinson and Ledbetter, 2012)), are effective in serving their intended purposes, none of them meets the CS annotation requirements perfectly. We need a tool that can help in sequence annotating in a way that can report the time needed for annotators to get their tasks done, manage number of annotator teams, enable quality control measures and annotation statistics, and assign some initial tags to some tags automatically (e.g. punctuation, URL, emoticon, etc.)

Our tool is most similar to the annotation tool for the COLABA project (Diab et al., 2010); (Benajiba and Diab, 2010)(Benajiba and Diab, 2010; Diab et al., 2010). We specifically emulate the annotator management component in the COLABA annotation tool. Although, the code switching annotation task and manual diacritization of Standard Arabic text task are completely different tasks, the MANDIAC tool (Obeid et al., 2016), which used for diacritization annotation task, has a similar annotator management component to the WASA management component. However, the technologies used in both management components are different. For instance, WASA uses PostgreSql database to store content, while MANDIAC uses a JSON blob to store content. Two other comparable tools to ours are WebANNO (Yimam et al., 2013) and SWAT (Samih et al., 2016). They both use the latest available technologies to perform a number of linguistic annotation types. The SWAT tool is a web-based interface for annotating tokens in a sequence with a predefined set of labels. The main advantages of this tool are the simplicity of its use and instal-

lation as it only requires a modern web browser and minimum server-side requirements to get the tool work. The WebANNO tool is also a web-based tool that offers wide range of linguistic annotations tasks, e.g., named entity, dependency parsing, co-reference chain identification, and part-of speech annotation.

However, both systems SWAT and WebANNO lack of some functionalities and features that can simplify and speed up the annotation task for our purposes. In the SWAT system for example, there is no support for user roles. Therefore, some tasks such as managing the number of annotators, monitoring the progress of the annotators, assigning tasks given to the annotators, and ensuring the quality of the submitted annotation are difficult to handle or manage with only one user type. Moreover, both systems do not have the option that can provide initial automatic tagging for named entities (NE), Latin words, URL, punctuation, number, diacritics, emoticon, and speech effects tokens. We noticed that tagging these tokens automatically increases the speed of the annotation substantially. Finally, unlike both systems, our system can seamlessly integrate with language specific data preprocessing tool to streamline raw data cleaning and preparation.

3. System Architecture

WASA is a typical three-tier web-based application. The platform is divided into three tiers, each with a specific function. The first tier is a data tier that saves all metadata in PostgreSql database in addition to both the annotated and raw data files. All this data is stored on a file server. The second tier is a logical tier. It contains PHP scripts that interact with an Apache web server. It is responsible for all functionalities provided by the system to the different types of users. All requests are sent by the web server to the PostgreSql database server through a secured tunnel. The third and last tier is the presentation tier. It is browser independent, which enables accessing the system from many different clients. It provides an intuitive GUI tailored to each user type. This architecture design allows multiple annotators to work on various tasks simultaneously. On the other hand, WASA allows the admin user to manage and handle a single central database. The system can handle multiple encodings allowing for multilingual processing. Figure-1 gives a high level overview of the tool's architecture.

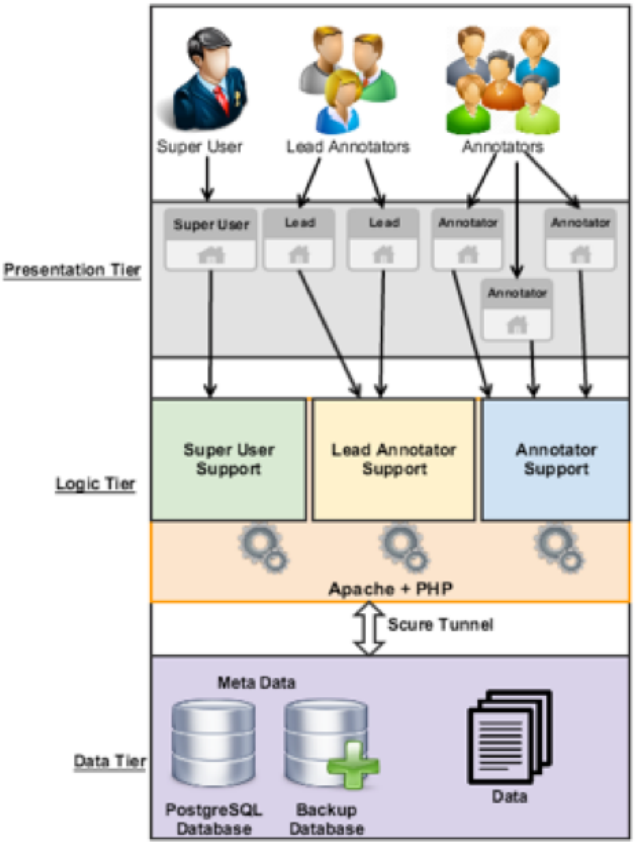

Figure 1: System Architecture

4. Types of Users

Three types of users have been considered in WASA design: Annotator, Lead-Annotator, and Super-User. Each one of these user types is given and provided with different kinds of permissions, functionalities, and privileges in order to fulfill their tasks.

4.1. Annotators

Annotators are provided the following functionalities: **1-** access assigned tasks; **2-**annotate the assigned tasks; **3-** submit annotation; **4-** check the time needed to submit one unit, e.g., post, or tweets; **5-** check the grade of the submitted work; **6-** re-annotate the rejected tasks (by rejected we mean when the annotator received a "No Pass" as a grade on their annotation task); and, **7-** save work and continue it in a later session.

Figure-2 shows an example of the annotation screen. The words of the posts or tweets that need to be annotated will be displayed as clickable units. When clicked, a pop-up screen appears to allow the annotator to choose the proper tag. To increase the speed of the annotation process, some of the words, like Named Entities and punctuations, will have an initial tag assigned automatically as part of a preprocessing step. However, the annotator is allowed to change the initial tag if he/she finds words annotated with a wrong tag. The interface uses color-coding to reflect useful information and status. For example, 'named entities' will be displayed in purple color, while Other tagged categories such as Latin, URL, punctuation, digits, diacritics, emoticons, sound effects will be displayed in the orange color. Words already annotated will be displayed in blue while words that are yet to be annotated appear in black. Figure-3 shows an example of some of the assigned tasks with information about the tasks that have been already submitted (e.g, number of annotated words, speed of annotation, path of the raw file)

4.2. Lead Annotator

For each dialect/language, there is one lead annotator only. Each lead annotator has the following functions: **1-** Annotator management, e.g., create, edit and delete annotator accounts; **2-** Tasks management; **3-** Monitor status and progress; **4-** Review and grade annotators' work; and **5-**

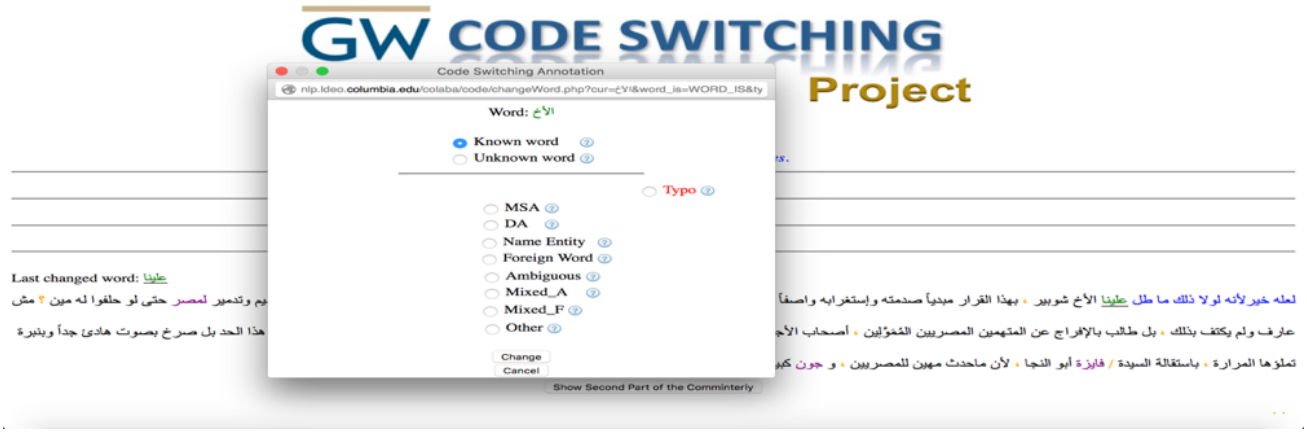

Figure 2: Annotation Screen

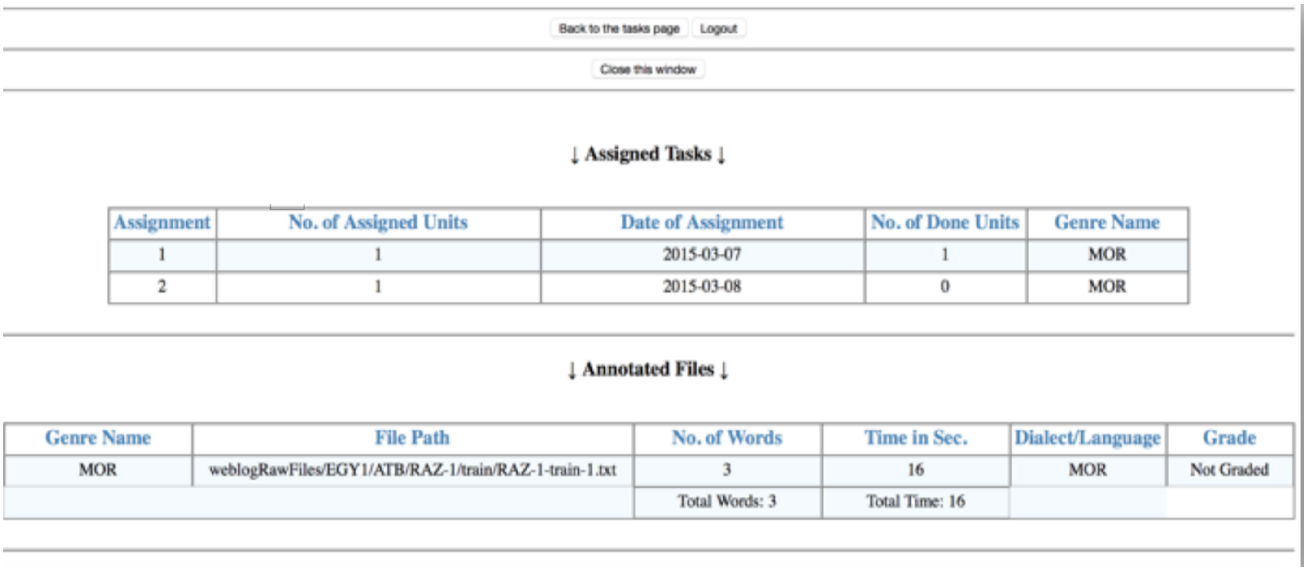

Figure 3: An example of the annotator's "Check-Status" screen

Produce quality measures like inter-annotator agreement. The system enables lead annotators to reject submitted work that does not meet the assessment criteria and add comments and feedback for the annotators to re-annotate rejected work.

4.3. Super User

There is only one Superuser account in WASA for all dialects/languages. The Superuser functions include: **1-** Database management and maintenance; **2-** Lead annotators management, **3-** Annotators management, **4-** Monitor the overall performance of the system; and **5-** Manage annotation data imports and exports.

5. Data Preprocessing and Input and Output Format

The system has the ability to integrate with language-specific date preprocessing scripts to streamline raw data cleaning and preparation. For example, for cleaning process (step-1) the system integrates the Smart Preprocessing (Quasi) Language Independent tool (SPLIT) (Al-Badrashiny et al., 2016) to handle the encoding issues (i.e, Change the character encoding to UTF8). Moreover, for the Dialectal Arabic (DA) and Modern Standard Arabic (MSA) language pair (step-2), the system integrates with the Automatic Identification of Dialectal Arabic (AIDA2) tool (Al-Badrashiny et al., 2015) to provide initial automatic tagging for named entities (NE), Latin words, URL, punctuation, number, diacritics, emoticon, and speech effects tokens. Figure-2 illustrates an example of a commentary with some pre-annotated tokens. Named entities tokens are colored purple, while punctuation and numbers are colored with orange. Both preprocessing and cleaning steps are performed offline. The Super User is the user responsible for preparing the data for annotation. Figure-5 shows the cleaning and preprocessing steps. The output file is written in a simple XML format as shown in Figure-4. The XML file includes all meta-data related to the annotation file such as the annotated, sentence id, task id, language, user id, word id, actual word, annotation tag, ...etc. The output XML is customizable. The superuser can choose what metadata to be included in the XML output file.

Our system is able to handle different types of genres such as Twitter, commentaries, conversations, or discussion fo-

rum data. Accordingly, WASA is quite robust as it is able to handle a variety of data genres and formats. For example, if the data comes from Twitter, then information like tweet id and user id needs to be preserved along with the annotation tags. If the genre of the data is discussion forums, information such as post order in the context of a conversation thread along with the names of the people who are involved in the conversation are maintained.

6. Database Design

WASA system uses a relational database to manage, handle and store all meta-data. The data stored is categorized as follow:

6.1. Profiling information

It saves information about all registered users of the system including their roles (i.e. annotator, lead annotator or superuser), login information as well as the dialect and languages for each one of them. Moreover, It contains information about different languages/dialects used in the project.

6.2. Annotation Information

This is the core part of WASA's database. It includes all meta-data related to the annotation tasks such as the number of tasks assigned to each annotator, actual annotations completed by each annotator, and temporarily saved annotations.

6.3. Assessment Information

This contains information about **1)** Task-Annotator assignment: it includes the tasks assigned to each annotator and the number of tasks that have already been annotated and submitted, the number of assigned units (tweets, posts) per each task, genre type, percentage of overlapping units (tweets, posts) shared among annotators to ease the process of calculating inter-annotator agreement, etc.; **2)** Annotator-Units assignment: It includes information about each unit (post, tweet) that is assigned to the annotators such as post/tweet-id, user-id, genre-id, task-id, path of the assigned file; Finally **3)** Language-Unit assignment: It includes information about the language/dialect id for each unit.

7. Quality Control Measures

WASA has built-in functionalities that can help in managing the inter-annotator agreement (IAA) measures for different task and report performance statistics. The lead annotator is able to specify the percentage of data annotation overlap between the annotators per task and the system manages to distribute the data and calculate the IAA. Moreover, WASA generates tag distribution, the number of annotated tokens, expected time needed to finish each assigned task, and much other quality management crucial statistics.

8. Current Status

We have tested the tool for annotation on Arabic MSA and dialectal data, Chinese-English, Spanish-English, and Hindi-English. The IAA for our the Arabic annotated data

```
<root>
  <document>
    <sentences>
      <sentence id="1">
        <tokens>
          <token id="1">
            <word>ولا</word>
            <Task>CSAnnot</Task>
            <UserID>2221</UserID>
            <language>ArabicMSA</language>
            <CharacterOffsetBegin>0</CharacterOffsetBegin>
            <CharacterOffsetEnd>3</CharacterOffsetEnd>
            <Annotation>MSA</Annotation>
            <AtomaticAnnot>No</AtomaticAnnot>
          </token>
      </sentence>
     <sentences>
  </document>
</root>
```

Figure 4: A sample of an output file

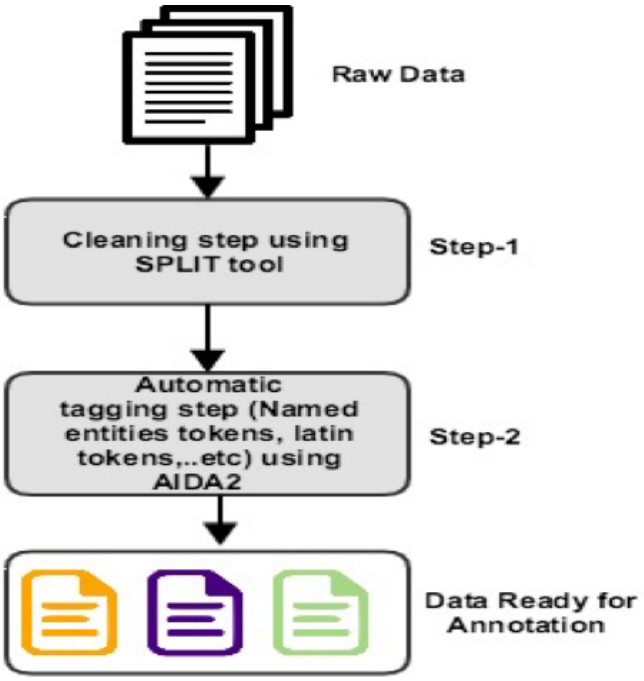

Figure 5: Preprocessing and Cleaning Steps

is ranged between 92% and 97%. Moreover, a small portion of the Code-Switching data that was released in (Diab et al., 2016) was used to test the performance of WASA. We noticed that the annotators' speed has increased substantially when we assign initial tags to some tags automatically (e.g. punctuation, URL, emoticon, ...etc.). The average time for annotating a full tweet was $\sim$ 40 seconds without using SPLIT tool (Al-Badrashiny et al., 2016), but after assigning the initial tags using the SPLIT tool, the average time for annotating a full tweet became $\sim$ 27 seconds. This results in saving much of the effort in annotating these tags.

9. Conclusion

We gave a detailed overview of our annotation system WASA. We have shown that WASA allows multiple annotator teams to work on various tasks simultaneously. Also,

we have seen that using the SPLIT tool to annotate some specific tokens automatically has helped in saving the effort and time spent by annotators. Moreover, the annotation quality of these tokens is very high. We will keep updating and modifying the current functionalities of the system as per different users type feedback. Also, we plan to add more functionality that can help in enhancing the speed, quality, and the efficiency of the CS annotation.

10. Acknowledgements

We would like to thank Mahmoud Ghoneim for his invaluable suggestions and support in the development of WASA. Also, We would like to acknowledge the useful comments by the three anonymous reviewers who helped in making this publication better presented.

11. Bibliographical References

Al-Badrashiny, M., Elfardy, H., and Diab, M. T. (2015). Aida2: A hybrid approach for token and sentence level dialect identification in arabic. In *CoNLL*, pages 42–51.

Al-Badrashiny, M., Pasha, A., Diab, M. T., Habash, N., Rambow, O., Salloum, W., and Eskander, R. (2016). Split: Smart preprocessing (quasi) language independent tool. In *LREC*.

Aswani, N. and Gaizauskas, R. (2009). Evolving a general framework for text alignment: Case studies with two south asian languages. In *Proceedings of the International Conference on Machine Translation: Twenty-Five Years On, Cranfield, Bedfordshire, UK, November*.

Aziz, W., Castilho, S., and Specia, L. (2012). Pet: a tool for post-editing and assessing machine translation. In *LREC*, pages 3982–3987.

Benajiba, Y. and Diab, M. (2010). A web application for dialectal arabic text annotation. In *Proceedings of the lrec workshop for language resources (lrs) and human language technologies (hlt) for semitic languages: Status, updates, and prospects*.

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Ursu, C., Dimitrov, M., Dowman, M., Aswani, N., Roberts, I., Li, Y., et al. (2009). *Developing Language Processing Components with Gate Version 5:(A User Guide)*. University of Sheffield.

Diab, M., Habash, N., Rambow, O., Altantawy, M., and Benajiba, Y. (2010). Colaba: Arabic dialect annotation and processing. In *Lrec workshop on semitic language processing*, pages 66–74.

Diab, M., Ghoneim, M., Hawwari, A., AlGhamdi, F., Al-Marwani, N., and Al-Badrashiny, M. (2016). Creating a large multi-layered representational repository of linguistic code switched arabic data. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may. European Language Resources Association (ELRA).

Dickinson, M. and Ledbetter, S. (2012). Annotating errors in a hungarian learner corpus. In *LREC*, pages 1659–1664.

Kahan, J., Koivunen, M.-R., Prud'Hommeaux, E., and Swick, R. R. (2002). Annotea: an open rdf infrastructure for shared web annotations. *Computer Networks*, 39(5):589–608.

Obeid, O., Bouamor, H., Zaghouani, W., Ghoneim, M., Hawwari, A., Alqahtani, S., Diab, M., and Oflazer, K. (2016). Mandiac: A web-based annotation system for manual arabic diacritization. In *The 2nd Workshop on Arabic Corpora and Processing Tools 2016 Theme: Social Media*, page 16.

Samih, Y., Maier, W., and Kallmeyer, L. (2016). Sawt: Sequence annotation web tool. *EMNLP 2016*, page 65.

Vargas-Vera, M., Motta, E., Domingue, J., Lanzoni, M., Stutt, A., and Ciravegna, F. (2002). Mnm: Ontology driven semi-automatic and automatic support for semantic markup. In *International Conference on Knowledge Engineering and Knowledge Management*, pages 379–391. Springer.

Yimam, S. M., Gurevych, I., Eckart de Castilho, R., and Biemann, C. (2013). Webanno: A flexible, web-based and visually supported system for distributed annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 1–6, Sofia, Bulgaria, August. Association for Computational Linguistics.