iAppraise: A Manual Machine Translation Evaluation Environment Supporting Eye-tracking

Ahmed Abdelali, Nadir Durrani, Francisco Guzmán
Qatar Computing Research Institute
Hamad Bin Khalifa University
Doha, Qatar
{aabdelali,ndurrani,fguzman}@qf.org.qa

Abstract

We present *iAppraise*: an open-source framework that enables the use of eye-tracking for MT evaluation. It connects Appraise, an open-source toolkit for MT evaluation, to a low-cost eye-tracking device, to make its usage accessible to a broader audience. It also provides a set of tools for extracting and exploiting gaze data, which facilitate eye-tracking analysis. In this paper, we describe different modules of the framework, and explain how the tool can be used in a MT evaluation scenario. During the demonstration, the users will be able to perform an evaluation task, observe their own reading behavior during a replay of the session, and export and extract features from the data.

1 Introduction

Evaluation is one of the difficult problems in Machine Translation (MT). Despite its clear drawbacks,[1] *human evaluation* remains the most reliable method to evaluate MT systems and track the advances in Machine Translation. Appraise is an open-source toolkit designed to facilitate the human evaluation of machine translation (Federmann, 2012). It has been adopted as the preferred tool in the WMT evaluation campaigns (Bojar et al., 2013), and thus, it is currently used by dozens of researchers.

According to the eye-mind hypothesis (Just and Carpenter, 1980) people cognitively process objects that are in front of their eyes. This has enabled researchers to analyze and understand how people perform certain tasks like reading (Rayner, 1998; Garrod, 2006; Harley, 2013). In recent times, eye-tracking has also been used in Machine Translation to identify and classify translation errors (Stymne et al., 2012), to evaluate the usability of automatic translations (Doherty and O'Brien, 2014), and to improve the consistency of the human evaluation process (Guzmán et al., 2015), etc. Furthermore, tracking how evaluators consume MT output, can help to reduce human evaluation subjectivity, as we could use evidence of what people *do* (i.e. unbiased reading patterns) and not only what they *say* they *think* (i.e. user-biased evaluation scores). However, the main limitation for the adoption of eye-tracking research has been the steep learning curve that is associated with eye-tracking analysis and the high-cost of eye-tracking devices.

In this paper, we present *iAppraise*: an open-source framework that enables the use of eye-tracking for MT evaluation, and facilitates the replication and dissemination of eye-tracking research in MT. First, it is designed to work with the increasingly popular, low-cost[2] eye-tracker *eyeTribe*. Secondly, it provides a set of tools for extracting and exploiting gaze features, which facilitate eye-tracking analysis. Lastly, it integrates fully with the Appraise toolkit, making it accessible to a larger audience.

Our setup allows to track eye-movements during the MT evaluation process. The data generated can be used to visualize a re-enactment of the evaluation session in real-time, thus providing useful qualitative insights on the evaluation; or to extract features for further quantitative analysis.

[1] It is subjective, expensive, time-consuming, boring, etc.

[2] It costs less than a hundred US dollars, and provides capabilities on par with previous generation eye-trackers.

The applications for this toolkit are multiple. Using reading patterns from evaluators could be a useful tool for MT evaluation: (*i*) to shed light into the evaluation process: e.g. the general reading behavior that evaluators follow to complete their task; (*ii*) to understand which parts of a translation are more difficult for the annotator; and (*iii*) to develop automatic evaluation systems that use reading patterns to predict translation quality. In an effort carried using this framework, we proposed a model to predict the quality of the MT output. Our results showed that reading patterns obtained from the eye-movements of the evaluators can help to anticipate the evaluation scores to be given by them. We found that the features extracted from the eye-tracking data (discussed in Section 2.6) capture more than just the fluency of a translation. Details of findings are reported in (Sajjad et al., 2016). In this paper, we describe the overall architecture of *iAppraise*: the communication modules, the user interface, and the analysis package.

2 iAppraise: Eye Tracking for Appraise

Appraise (Federmann, 2012) is an open-source toolkit,[3] used for for manual evaluation of machine translation output. However, it also allows to collect human judgments on a number of annotation tasks (such as ranking, error classification, quality estimation and post-editing) and provides an environment to maintain and export the collected data. The toolkit is based on the Django web framework that supports database modeling and object-relational mapping, and uses Twitter's Bootstrap as a template for the interface design.

iAppraise consist of a series of modules that extend Appraise to integrate eye-tracking from the eyeTribe[4] into the translation evaluation tasks. Below we briefly describe the architecture of the toolkit.

2.1 Overall Architecture

In Figure 1 we present the overall architecture of our toolkit. First, the *iAppraise Adapter* communicates directly with the EyeTribe eye-tracker through its API and propagates the gaze events to the *iAppraise UI (User Interface)*.

[3]Available at: `github.com/cfedermann/Appraise`

[4]http://dev.theEyeTribe.com/api/

The *iAppraise UI* module takes the gaze events, and translates their coordinates into local browser coordinates. It also converts all the textual material in the display into *traceable* objects, that can detect *Gaze* when a user is looking at them. Additionally, the module contains a view-task whose layout is optimized for the recognition of gaze events.

When a traceable object in *iAppraise UI* detects that a user is looking at it, it stores this information, augmented with UI details from the gaze data. This data is later stored in *iAppraise Model/DB* at the end of each evaluation session. Finally, the *iAppraise Analysis* module is designed to extract useful eye-tracking features from the generated data. These can be used for modeling or analysis.

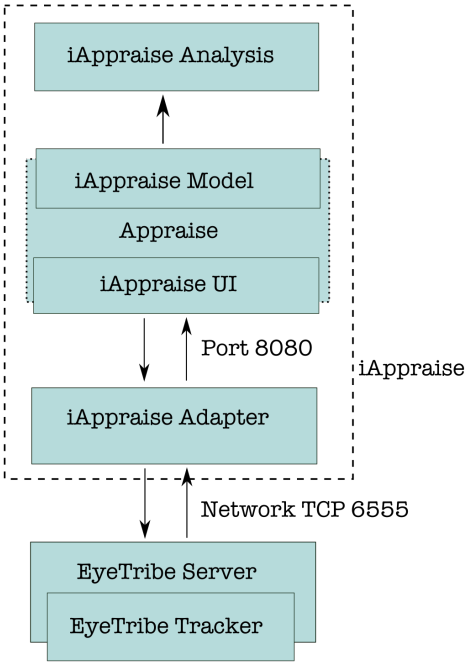

Figure 1: iAppraise architecture

2.2 iAppraise Communication Interfaces

The *EyeTribe* eye-tracker, running at 30Hz or 60Hz, broadcasts gaze data through a TCP port (EyeTribe default port 6555) using *JSON* (JavaScript Object Notation) formatted messages. The iAppraise Adapter employs two sockets, one that listens to the eye-tracker, and the other to pass the data to the *iAppraise* UI.

2.3 iAppraise User Interface

To facilitate the usage of eye-tracking data for machine translation evaluation, we added an evaluation task to the original Appraise.[5] This task has a layout and graphical elements, that have been optimized for the use of eye-tracking. The template has two main content regions: *Reference* and *Translation*. The task for this view requires to score the quality of a translation by comparing it to the provided reference. The annotator is required to use a slider (see Figure 2) to provide a score. In return, he/she gets feedback in the form of stars, that reflect how close his/her score is to an optional gold-standard score. In principle, the stars are part of a gamification strategy used to keep the evaluator engaged. If the gold standard scores are not be available this option can be turned off.

2.4 iAppraise Model/DB

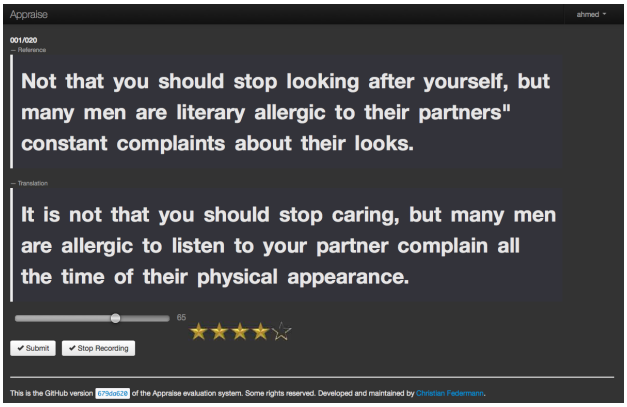

Figure 2: iAppraise Eye-Tracking Evaluation task layout with feedback. From top to bottom: Reference and Translation sentences, slider for scoring, and feedback in a form of stars.

To handle data flow and gather information resulted from the eye-tracking and user interaction, a new data model was added to Appraise. This data model stores the data received from the *iAppraise UI* into a database. Table 1 shows the different attributes and the description of the fields for the data recorded during an eye-tracking task.

[5] `appraise/templates/evaluation/eyetracking.htm`

Attribute	Description
Task attributes	
pscore	Eye-tracker precision at the time of the recording. Computed as the number of words observed in a random sample of words.
score	Score given by the user
Gaze attributes	
region	Active region where the gaze landed
gazex	Actual coordinate x of the gaze
gazey	Actual coordinate y of the gaze
data	JSON EyeTribe message[6]
Environment attributes	
scaling	The ratio of the (vertical) size of one physical pixel on the current display to the size of one device independent pixels(dips)
zoom	Window zooming level
scrollx	Number of horizontal pixels the current document has been scrolled from the upper left corner of the window
scrolly	Number of vertical pixels the current document has been scrolled from the upper left corner of the window
clientWidth	Window width
div0Height	Window height
innerHeight	The inner height of the browser window
outerHeight	The outer height of the browser window

Table 1: Description of attributes stored in the database.

2.5 Eyetacking Replay

The eye-tracking data collected during the evaluation session can be visualized as a re-enactment or replay. This allows to analyze the evaluator during the task, or to perform some basic troubleshooting. The replay highlights the background of words in the sequence that they were observed (See Figure 3 for demonstration).

2.6 iAppraise Analysis

The *iAppraise Analysis* module extracts useful features from the *iAppraise DB* that can be used to analyze the evaluation process or a train a prediction model. It consists of several auxiliary scripts that parse the data and extract features described below:

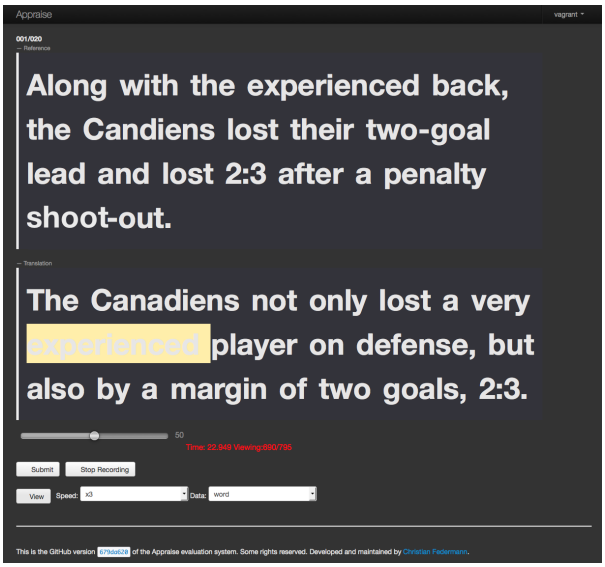

Figure 3: iAppraise Eye-Tracking task replay; words gets highlighted in the sequence that they were observed.

Jump features While reading text, the gaze of a person does not visit every single word, but advances in jumps called *saccades*. These jumps can go forward (*progressions*) or backward (*regressions*). We classify the word-transitions according to the direction of the jump and distance between the start and end words. For subsequent words n, $n+1$, this would mean a forward jump of distance equal to 1. All jumps with distance greater than 4 are sorted into a 5+ bucket. Additionally, we separate the features for reference and translation jumps. We also count the total number of jumps.

Total jump distance We aggregate jump distances[7] to count the total distance covered while evaluating a sentence. We count reference and translation distance features separately. Such information is useful in analyzing the complexity and readability of the translation.

Inter-region jumps While reading a translation, evaluators can jump between the translation and a reference to compare them. Intuitively, more jumps of this type could signify that the translation is harder to evaluate. Here we count the number of reference↔translation transitions.

[7]Jump count and distance features have also shown to be useful in SMT decoders (Durrani et al., 2013).

Dwell time The amount of time a person fixates on a region is a crucial marker for processing difficulty in sentence comprehension (Clifton et al., 2007) and moderately correlates with the quality of a translation (Doherty et al., 2010). We count the time spent by the reader on each particular word. We separate reference and translation features.

Lexicalized features The features discussed above do not associate gaze movements with the words being read. We believe that this information can be critical to judge the overall difficulty of the reference sentence, and to evaluate which translation fragments are problematic to the reader. To compute the lexicalized features, we extract streams of reference and translation lexical sequences based on the gaze jumps, and score them using a tri-gram language model. Let $R_i = r_1, r_2, \ldots, r_m$ be a sub-sequence of gaze movement over reference and there are $R_1, R_2, \ldots, R_n$ sequences, the lex feature is computed as follows:

$$lex(R) = \sum_{i}^{n} \frac{\log p(R_i)}{|R_i|}$$

$$p(R_i) = \sum_{j}^{m} p(r_j | r_{j-1}, r_{j-2})$$

The normalization factor $|R_i|$ is used to make the probabilities comparable. We also use unnormalized scores as additional feature. A similar set of features $lex(T)$ is computed for the translations. All features are normalized by the length of the sentence.

In a related effort, we used the above features to predict the quality scores given by an evaluator. More details on the model and how effective each of the features were, please refer to Sajjad et al. (2016).

3 iAppraise Demonstration Script

iAppraise demonstration will allow the users to experiment with the tool and the eye tracking device. The users will be able to perform an evaluation task, observe a replay of their own eye movements, and to export their gaze data. We will also demonstrate the basic functioning for additional tools and scripts. This includes using the exported data to extract features and information about the evaluation task.

The iAppraise server is available as an open-source project, and can also be downloaded as an already-configured virtual machine that can be deployed on any environment.

4 Conclusion

In this paper, we presented *iAppraise*, a framework to provide eye-tracking capabilities to directly Appraise. Here we described the different components that make up the framework. The main goal of the framework is to provide a tool that lowers the entry-level bar to using eye-tracking in the MT community. *iAppraise* has several advantages: (*i*) it connects low-cost eye-trackers to an open-source MT analysis platform; and (*ii*) it provides a set of analysis tools that allow the use of the gaze information effortlessly. We expect that in the future, more researchers will adopt *iAppraise* to explore the human consumption of text in other NLP tasks.

References

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria, August. Association for Computational Linguistics.

Charles Clifton, Adrian Staub, and Keith Rayner. 2007. Eye movements in reading words and sentences. *Eye movements: A window on mind and brain*, pages 341–372.

Stephen Doherty and Sharon O'Brien. 2014. Assessing the usability of raw machine translated output: A user-centered study using eye tracking. *International Journal of Human-Computer Interaction*, 30(1):40–51.

Stephen Doherty, Sharon O?Brien, and Michael Carl. 2010. Eye tracking as an MT evaluation technique. *Machine translation*, 24(1):1–13.

Nadir Durrani, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013. Can Markov models over minimal translation units help phrase-based SMT? In *Proceedings of the Association for Computational Linguistics (ACL'13)*, Sofia, Bulgaria.

Christian Federmann. 2012. Appraise: an open-source toolkit for manual evaluation of mt output. *The Prague Bulletin of Mathematical Linguistics*, 98:25–35.

Simon Garrod. 2006. Psycholinguistic research methods. *The encyclopedia of language and linguistics*, 2:251–257.

Francisco Guzmán, Ahmed Abdelali, Irina Temnikova, Hassan Sajjad, and Stephan Vogel. 2015. How do humans evaluate machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 457–466, Lisbon, Portugal, September. Association for Computational Linguistics.

Trevor A Harley. 2013. *The psychology of language: From data to theory*. Psychology Press.

Marcel A Just and Patricia A Carpenter. 1980. A theory of reading: from eye fixations to comprehension. *Psychological review*, 87(4):329.

Keith Rayner. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin*, 124(3):372.

Hassan Sajjad, Francisco Guzmán, Nadir Durrani, Ahmed Abdelali, Houda Bouamor, Irina Temnikova, and Stephan Vogel. 2016. Eyes Don't Lie: Predicting Machine Translation Quality Using Eye Movement. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, California, June. Association for Computational Linguistics.

Sara Stymne, Henrik Danielsson, Sofia Bremin, Hongzhan Hu, Johanna Karlsson, Anna Prytz Lillkull, and Martin Wester. 2012. Eye tracking as a tool for machine translation error analysis. In *LREC*, pages 1121–1126.