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-cost2 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.

---

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

2It 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, used 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 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).

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.

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. 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.

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

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 to train a prediction model. It consists of several auxiliary scripts that parse the data and extract features described below:

---

5appraise/templates/evaluation/eyetracking.htm
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.

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 un-normalized 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.

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


