Proceedings of the 10th Workshop on the Representation and Processing of Sign Languages (sign-lang@LREC 2022), pages 159–164
Language Resources and Evaluation Conference (LREC 2022), Marseille, 20-25 June 2022

Towards Semi-automatic Sign Language Annotation Tool: SLAN-tool

Medet Mukushev[1], Arman Sabyrov[1], Madina Sultanova[1], Vadim Kimmelman[2], Anara Sandygulova[1]
[1]Department of Robotics and Mechatronics, School of Engineering and Digital Sciences
Nazarbayev University, Nur-Sultan, Kazakhstan
[2]Department of Linguistic, Literary, and Aesthetic Studies
University of Bergen, Bergen, Norway
{mmukushev, arman.sabyrov, madina.sultanova, anara.sandygulova}@nu.edu.kz
vadim.kimmelman@uib.no

Abstract

This paper presents a semi-automatic annotation tool for sign languages namely SLAN-tool. The SLAN-tool provides a web-based service for the annotation of sign language videos. Researchers can use the SLAN-tool web service to annotate new and existing sign language datasets with different types of annotations, such as gloss, handshape configurations, and signing regions. This is allowed using a custom tier adding functionality. A unique feature of the tool is its automatic annotation functionality which uses several neural network models in order to recognize signing segments from videos and classify handshapes according to HamNoSys handshape inventory. Furthermore, SLAN-tool users can export annotations and import them into ELAN. The SLAN-tool is publicly available at https://slan-tool.com.

Keywords: Sign language, sign language annotation, multimedia annotation tools

1. Introduction

Most of the existing sign language datasets use sentence-level translations and glossing for annotation of sign language (SL) data. Glossing is a simplified notation system used to transcribe sign language with written words in a spoken language. Additionally, some corpora can also be enhanced with annotations of handshape configurations, mouthing cues and other non-manual markers, or keypoint locations of the body. However, such extra annotations are not common for all datasets. As sign languages make use of the rich visual modality by employing hand gestures, facial expressions, body and head orientation and movement, this information is lost if only textual annotations are provided.

In contrast to automatic speech recognition, no computational tools exist to conduct semi-automatic sign language annotation. As a result, annotating sign language corpora is a time-consuming manual operation. Furthermore, there are no widely accepted annotation standards. Bragg et al. (2019) highlight the lack of a standardized annotation system and annotation granularity. As a result, experts are unable to merge sign language datasets. It is vital to provide uniform annotations as input for Natural Language Processing (NLP) and Machine Translation (MT) systems in order to train accurate and dependable models (Bragg et al., 2019). Because there is no technology to automatically tag or annotate sign language data in the quality required for linguistic annotation, corpus developers have been compelled to manually annotate the data. (Kopf et al., 2021).

There is a need for a web-based program, that provides the required flexibility to automate accurate, customizable analysis and data annotation. To address this, we developed a semi-automatic tool tailored for annotation of sign language videos. We propose the Sign Language ANnotation-tool (SLAN-tool) that semi-automatically divides videos into segments with active signing, identifies handshape configurations, and enables users to edit and export annotations. Our main contributions are as follows:

- SLAN-tool provides a web-based service for manual and semi-automatic SL annotation. The tool is freely available at https://slan-tool.com.
- We developed a neural network model to find segments of active signing in longer videos. This can help to work with shorter versions of the video and decrease annotation time.
- SLAN-tool provides extended handshape configuration classification model with more than 80 handshape classes. For the ease of use, they are divided into categories according to HamNoSys (Schmaling and Hanke, 2001) notation system.

2. Related work

There are various video annotation software packages available that are often used for sign language annotation.

ELAN (Wittenburg et al., 2006) is a tool for annotating audio and video recordings. A user can add an extensive list of textual comments to audio and/or video recordings using ELAN. An annotation can be a phrase, a word, a gloss, a comment, a translation, or a description of anything seen in the media. Annotations could be produced on several layers, known as tiers, that could be integrated hierarchically. An annotation might be time-aligned to the media or link to

other annotations that already exist. Annotation output is Unicode text, and annotation documents are saved in XML format (EAF). ELAN is free and open source (GPLv3), and it may be installed on Windows, macOS, and Linux. Crasborn and Sloetjes (2008) enhanced it specifically for sign language corpora annotation.
Neidle et al. (2001) proposed SignStream, which is aimed to make linguistic annotation and analysis of video data easier. It may be used to annotate handshapes and show non-manual characteristics. SignStream is only available for MacOS versions and is released under the MIT license.
iLex (Hanke and Storz, 2008) is a corpus and sign language lexicography analysis software that integrates characteristics from empirical sign language lexicography and sign language dialogue transcription. It assists the user in constructing an integrated vocabulary while working on the transcription of a corpus and provides a number of additional features. MacOS binaries for iLex are available for installation.
There are several works focusing on automatic annotation of Sign Languages. Chaaban et al. (2021) presented an automatic annotation system for face and body annotations such mouthing, head direction, and sign position. Furthermore, their system was able to automatically splits signs based on hand movements. De Coster et al. (2019) developed a gloss suggestion system based on OpenPose (Cao et al., 2019) keypoint extraction library. It provides annotation suggestion for a selected video clip by showing top 5 predictions.

3. Methodology

First, we discuss the user requirements collecting approach that was utilized to acquire system needs. Following that, we will go into the system design and user interfaces that were created based on the requirements that were obtained. Finally, we cover neural network models that are employed for automated annotation of signature segments and categorization of handshapes.

3.1. User requirements

We began by studying and comparing current sign language annotation tools. There are various options, the most common of which is ELAN. ELAN includes a lot of features. Simultaneously, it has a severe learning curve for first-time users.
Following preliminary study, the goal of this project was clear: to present researchers with a specialized tool for semi-automatic annotation of sign language recordings. The major aims were to provide a web-based interface for the annotation tool and semi-automatic annotation generating modules. We conducted interviews with potential users of the system, including sign language researchers and data annotators, to get high-level abstract needs. The following user needs were gathered:

- to upload and play the selected video on the main page;
- to send uploaded videos to the annotation generation module for processing;
- to view generated annotations in relevant tiers on main page;
- to adjust and update generated annotations (change predicted class, adjust segmentation boundaries, etc.);
- to add custom tiers for annotation if needed;
- to export and import generated annotations (JSON, CSV, ELAN format);
- to share results of the annotation with other people.

3.2. User Interface (UI) and functionality

The annotation tool's UI consists of the main page and supplementary pop-up windows with menu choices. The main page is divided into four sections: control functions, video player, annotation tiers, and supplementary visualization.

1. The control functions area needs to have the following buttons: Upload video, Process video, Export/import annotation file, Save project, Share project, Annotate.
2. The video player area needs to display the uploaded video and a timeline underneath it.
3. The additional visualization area needs to display information that is not suitable for tiers.
4. The annotations tiers area needs to display a predefined list of tiers such as translation, gloss, right handshape, left handshape. Additional tiers can be added by users when needed.

3.3. System design

System design requirements are more thorough definitions of the functions, services, and operational limitations of a software system. The system requirements specify precisely what should be implemented. To make it more convenient for users, we decided to create a web-based solution. It assists in the avoidance of issues associated with the installation of certain software libraries and the availability of computing resources. The annotation tool was decided to be accessible via preferred web browsers and to feature an easy-to-use UI. The cloud servers undertake automatic annotation of the videos. Figure 1 shows an overview of the Sign Language Annotation tool's architecture. Figure 2 shows the proposed User Interface for SLAN tool.

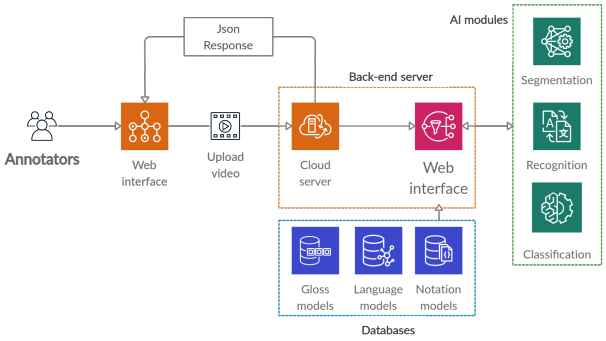

Figure 1: Overview of the Sign Language Annotation tool's Web service.

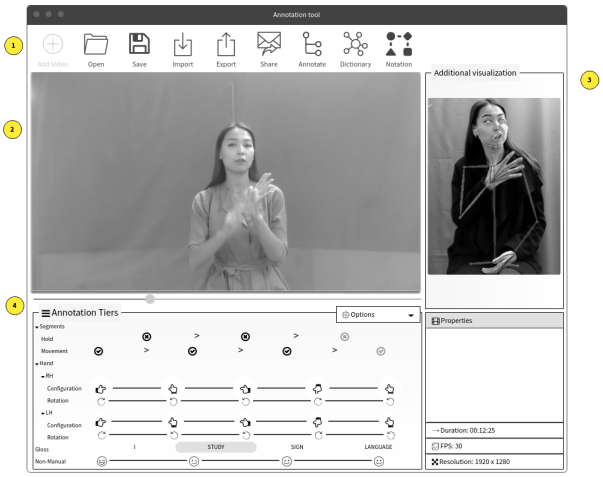

Figure 2: Proposed User interface for annotation tool.

3.4. Neural network models

3.4.1. Signing segmentation model

The fundamental concept is to assist annotators in automatically locating and working with active areas of the video. This can help to improve the efficiency and speed of annotation for long sign language videos. It was decided to identify video segments when signing happens i.e. a signing segment. This task may be compared to an action recognition task. To this end, the segmentation task involves recognizing frame boundaries in videos to separate them into meaningful units. These units can be a series of glosses or subtitle-units matched to sign language videos. To train detection algorithms, both techniques require annotated sets of videos.

3.4.2. Handshapes classification model

Handshape images gathered from the large handshape dataset (Koller et al., 2016) are divided by HamNoSys annotation, yielding 84 classes and 101 098 samples in total. On a test set, the training strategy on all classes performed poorly in terms of generalization. As a result, we devised a method that first determines the category of the handshape image. A category is a set of handshape configuration classes that are comparable to one another. After identifying the category, another model is utilized to determine the class handshape inside the category. We tried numerous tactics in order to find the optimal one that outperformed on the test set.

4. Implementation

The SLAN-tool was built with a variety of Open Source libraries and software technologies. The SLAN-source tool's code will be published under the BSD-2 clause license.

4.1. Annotation tool

User interface is implemented with HTML5, CSS3, JS, JQuery and Bootstrap library. Back-end processing is implemented with Python programming language, Django framework, Flask machine learning framework, and PostgreSQL database. AWS S3 is used as a cloud server. Networking is performed with Gunicorn and Nginx.

4.2. Classification Models

Sign segmentation and Handshape classification models have been pre-trained using the TensorFlow (Abadi et al., 2016) and PyTorch (Paszke et al., 2019) machine learning frameworks to recognise and classify sign language videos automatically.

4.2.1. Sign Language Segmentation

In order to train our model we divide sign language videos into three categories: signing-start, signing-end, no-signing segments. For training the model we have extracted videos for each category from three different datasets, KRSL (Imashev et al., 2020), WLASL (Li et al., 2020) and Dicta-Sign–LSF–v2 (Belissen et al., 2020), which were manually labeled.

We used R(2+1)D (Tran et al., 2018) action recognition model, which is highly accurate and at the same time significantly faster than other approaches. Its accuracy comes in large parts from an extra pre-training step which uses 65 million automatically annotated video clips. Its speed comes from simply using video frames as input. Many other state-of-the-art methods require optical flow fields to be pre-computed which is computationally expensive.

4.2.2. Handshape Configuration Classification

We implemented several strategies to discover the best one that has better performance over a test set.

First, every 4 neighbour classes were merged as shown in Figure 3.A, and the training process included 36 categories. The model is fine-tuned by changing hyperparameters and as a result, it is under-fitting. It showed poor performance on the training set, so both training and validation accuracy was not higher than 30%.

Second, we consolidated classes by HamNoSys rows, thus each category in this strategy has 11–22 classes. Figure 3.B presents an illustration of that strategy. This approach also did not show promising results. Both training and validation accuracy did not exceed 40%.

Figure 3: Different strategies for handshape categories based on HamNoSys Handshape Chart (Hanke, 2010)

	Sub-category classes (1 - 6)	Total images	Train images	Validation images
1	6	59056	47243	11813
2	5	5758	3998	998
3	6	9433	7543	1890
4	1	154	N/A	N/A
5	6	3951	3159	792
6	3	8416	6732	1684
7	4	4916	3931	985
8	3	6825	5460	1365
9	2	2487	1989	498

Table 1: Sub-category classes of Categories

Results present high bias and low variance which are indicators of the under-fitting again. Since it could happen due to model simplicity, EfficientNet-B5 (Tan and Le, 2019) model architecture was replaced by EfficientNet-B7 (Tan and Le, 2019). Additional training for more time or epochs in this step also shows poor validation accuracy.

Finally, we come to the best way, of consolidating classes into categories presented in Figure 3.C. By this strategy, we start with training a model on 9 large categories. To improve the model accuracy optimizers, their learning rates, decay, and other hyper-parameters were carefully selected. Accordingly, only after the identification of the handshape category, we start training by sub-categories which are described in the first strategy. As it can be seen from Figure 3.D we have 1-6 sub-category classes inside each category (Table 1). Afterwards, the result of this step is a total of 8 models which give different sub-categories where each has 4 classes. Furthermore, each sub-category trained only on at most 4 classes. This approach demonstrates the best generalization from the beginning, while all previous ones have failed.

4.3. Demonstration

The main page consists of 4 areas: control functions, video player, annotation tiers, and an additional visualization. The annotations tiers area display a predefined list of tiers such as text and handshapes. There are buttons to add and remove custom tiers. On the right panel handshapes menu is shown when users work with a handshape annotation tier. Figure 4 shows current interface of the SLAN tool.

5. Usability testing

In order to conduct usability testing, we invited 3 sign language data annotators. The participants are experienced in using a web-based annotation tool SurdoBot (https://surdobot.kz) for gloss annotation of sign language videos. It is a custom built tool that was used to annotate short clips of KRSL dataset.

We performed 1 hour individual Zoom sessions in which the participants were asked to annotate short sign

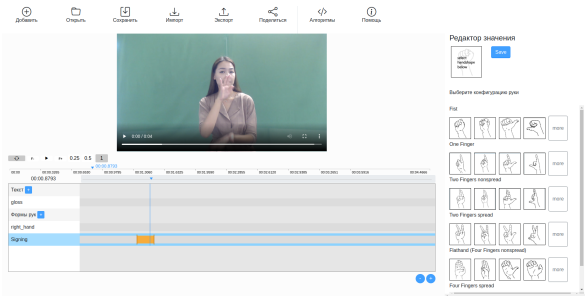

Figure 4: Current UI of the SLAN tool.

language video clips. We performed two test scenarios. First, we compared SLAN-tool to the service they used before for sign language data annotation. Next, after they got familiar with the SLAN-tool, we asked them to compare manual annotation to automatic annotation. Below are details of both scenarios.
The procedure of the usability test was specifying a task and asking the users to speak aloud their thinking process. The following tasks were specified:

- Could you please annotate the video by writing its translation in a written form?
- Could you please annotate regions of individual signs?
- Could you please annotate several handshape configurations?
- Since SLAN tool provides an automatic handshape annotation functionality, could you please launch it?
- What do you think about the layout and user interface in general?

Overall, the participants had some difficulties with adding tiers and gloss annotation for the first time. These issues were mainly because they had been using a simpler tool which had only one functionality. We have changed the instructions section by adding the "Help" button to the main menu. After that, when the participants had difficulties with any functions of the tool, they were able to quickly find instructions.
Another suggestion from most participants was to change the input method. We added options to directly enter annotations next to the selected segment. It helped to increase annotation speed and made the process more convenient.
Regarding the automatic annotation functionality, all the participants agreed that it makes annotations process easier and faster. After automatic annotation, the participants just needed to edit and adjust the selected segments only.

6. Discussion

The SLAN-main tool's goal is to provide a convenient functionality that does not require any further software installation. All users have to do is go to the website and upload their videos. The web service is freely accessible and does not involve the use of additional computing resources on the client's site. Currently, the service is hosted on an AWS dedicated server. The SLAN-tool can be used in conjunction with the ELAN-tool. It supports export and import in the same format as the ELAN software. For example, the SLAN-tool may be used to automatically annotate a sign language video and then export the results to ELAN for further annotations.
There are several use cases of the SLAN-tool:

- Automatic annotation: SLAN-tool can be used to automatically divide signing videos into shorter segments. Then for each segment the tool can identify handshape configurations and annotate them. Later these annotations can be exported to other tools such as ELAN for additional processing.
- Gloss notation: if the user needs to quickly annotate a sign language video it can be done in SLAN-tool by adding custom glossing tiers. When combined with the segmentation model, the annotation process takes shorter time as the user only needs to focus on active segments.

Currently, the main limitation is the computational resources available for the SLAN-tool. We are using a self-hosted server with 2 GPUs for video processing. For this reason, users have limitation on the duration of annotated videos. In future, we plan to migrate to a cloud-based server where researcher will be able to automatically annotate longer videos.

7. Conclusion

Our proposed tool automatically annotates some features in sign language videos and enables researcher to extend annotations for their datasets. With the help of SLAN-tool, researchers will have faster and cost effective annotation process.
SLAN-tool, as for now, has 2 models for automatic annotation: segment detection and handshape configuration classification. Other functionalities, such as hand orientation, location and movement are planned to be released.
Additionally, the tool will support automatic spotting of the most common signs (their detection and classification). Also, we will release all source codes, so that researchers can use tool on their computers if needed.

8. Acknowledgements

This work was supported by the Nazarbayev University Faculty Development Competitive Research Grant Program 2019-2021 "Kazakh Sign Language Automatic Recognition System (K-SLARS)". Award number is 110119FD4545.

9. Bibliographical References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016). TensorFlow: A System for Large-Scale Machine Learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, page 265–283, USA. USENIX Association.

Belissen, V., Braffort, A., and Gouiffès, M. (2020). Dicta-Sign-LSF-v2: Remake of a continuous French Sign Language dialogue corpus and a first baseline for automatic sign language processing. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6040–6048, Marseille, France, May. European Language Resources Association.

Bragg, D., Koller, O., Bellard, M., Berke, L., Boudreault, P., Braffort, A., Caselli, N., Huenerfauth, M., Kacorri, H., Verhoef, T., et al. (2019). Sign language recognition, generation, and translation: An interdisciplinary perspective. In *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*, pages 16–31.

Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S., and Sheikh, Y. A. (2019). OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Chaaban, H., Gouiffès, M., and Braffort, A. (2021). Automatic Annotation and Segmentation of Sign Language Videos: Base-level Features and Lexical Signs Classification. In *VISIGRAPP*.

Crasborn, O. and Sloetjes, H. (2008). Enhanced ELAN functionality for sign language corpora. In *Proceedings of the 3rd Workshop on the Representation and Processing of Sign Languages at LREC*, pages 39–43, Marrakech, Morocco. European Language Resources Association.

De Coster, M., Van Herreweghe, M., and Dambre, J. (2019). Towards automatic sign language corpus annotation using deep learning. In *Proceedings of the 6th Workshop on Sign Language Translation and Avatar Technology*.

Hanke, T. and Storz, J. (2008). iLex–A Database Tool for Integrating Sign Language Corpus Linguistics and Sign Language Lexicography. In *Proceedings of the 3rd Workshop on the Representation and Processing of Sign Languages at LREC*, pages 64–67, Marrakech, Morocco. European Language Resources Association.

Hanke, T. (2010). HamNoSys 4 Handshapes Chart. Drawings by H. Zienert, O. Jeziorski, A.Hanß.

Imashev, A., Mukushev, M., Kimmelman, V., and Sandygulova, A. (2020). A dataset for linguistic understanding, visual evaluation, and recognition of sign languages: The K-RSL. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 631–640.

Koller, O., Ney, H., and Bowden, R. (2016). Deep Hand: How to Train a CNN on 1 Million Hand Images When Your Data is Continuous and Weakly Labelled. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3793–3802.

Kopf, M., Schulder, M., and Hanke, T. (2021). Overview of Datasets for the Sign Languages of Europe, July.

Li, D., Rodriguez, C., Yu, X., and Li, H. (2020). Word-Level Deep Sign Language Recognition from Video: A New Large-Scale Dataset and Methods Comparison. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1459–1469.

Neidle, C., Sclaroff, S., and Athitsos, V. (2001). SignStream: A tool for linguistic and computer vision research on visual-gestural language data. *Behavior Research Methods, Instruments, & Computers*, 33(3):311–320.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA.

Schmaling, C. and Hanke, T. (2001). HamNoSys 4.0. *Interface definitions. ViSiCAST Deliverable D5-1.*

Tan, M. and Le, Q. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6105–6114. PMLR.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., and Paluri, M. (2018). A Closer Look at Spatiotemporal Convolutions for Action Recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6450–6459.

Wittenburg, P., Brugman, H., Russel, A., Klassmann, A., and Sloetjes, H. (2006). ELAN: a Professional Framework for Multimodality Research. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, Genoa, Italy. European Language Resources Association.