CS 231N - Project Milestone - American Sign Language Gesture Detection

Sean Konz
Stanford University
swkonz@stanford.ed

Jonathan Griffin
Stanford University
jgriffi2@stanford.edu

Brandon Huelga
Stanford University
bhuelga@stanford.edu

Abstract

Over 500,000 people in the United States and Canada use American Sign Language (ASL). In this project, we leverage various activity recognition models in an attempt to help ASL signers better engage with non-ASL speakers by developing a sign-translator. Previous research has made strides in developing classification systems for static signs and letters; in contrast, we aim to recognize phrases and words that use dynamic signs that utilize hand and arm movement. We use the American Sign Language Lexicon Video Dataset to train a number of models to achieve this task, and use top-5 classification accuracy as an evaluation metric. We compare the performance of 3D Convolutional Networks and more traditional Recurrent Neural Networks on this dataset and find that simplified models and sparse data contribute to the difficulty of this task. We leverage simplified data representations in order to efficiently train algorithms for this task, and then propose future approaches that may have greater success. Overall, our project demonstrates the difficulty in recognizing human activities in videos with hundreds of frames and potential classes.

1. Introduction

American Sign Language (ASL) is used by over 500,000 people in the US and Canada as a means for communication. Additionally, approximately 90,000 college students in the U.S. started learning ASL in 2009, and this number has only increased since then. With such a stark increase in ASL signers, the development of an automated ASL-to-English translator would allow ASL signers to better communicate with non-ASL speakers. This task is non-trivial since visual recognition and understanding are far less advanced than current speech-to-text and Natural Language Processing systems. ASL uses continuous, dynamic signing to formulate sentences, and in general, the signs most commonly employed are not static hand maneuvers, but are continuous, fluid hand movements, each of which flows into the next sign. Thus, the task of conducting speech recognition for ASL signs is reliant on classifying multiple image frames of a sign being made at once. This task shares characteristics of traditional activity recognition, which has seen increased research interest since the dawn of the "deep learning" revolution.

Activity recognition tasks are a natural stepping stone from classification of still images due to the integration of a temporal component in the recognition task [9]. The temporal dimension of videos provides an additional clue for recognition since some actions can be easily recognized based on motion information (i.e. ASL signs). However, the task is much more computationally demanding since each video can contain hundreds of image frames that need to be processed individually, and the addition of a temporal component is difficult to fully utilize [6].

Popular approaches to activity recognition generally center around the use of Convolutional Neural Nets (CNNs) and Recurrent Neural Nets (RNNs). However, despite the success of CNNs in still image classification, CNN use in activity recognition has not been met with the same degree of success. This may be in part due to the fact that current datasets are relatively small and noisy in comparison to image classification datasets, such as ImageNet. Additionally, activity recognition in videos often faces challenges of variation of motion and viewpoint, and thus models might require more training data than that of image classification [9]. Another important reason for this accuracy discrepancy is that current CNN architectures are not able to take full advantage of temporal information and their performance is consequently dominated by spatial recognition [3]. Furthermore, when CNNs are expanded into three dimensions (3D-CNN) their ability to utilize the temporal dimension is increased, but their complexity is significantly increased. RNNs are better suited for this task, however, an RNN operating on flattened frames from a video is not able to utilize the spatial features of an image.

With these limitations in mind, we attempt to develop an ASL-to-English translator which takes an activity recognition approach to translating individual signs into their equivalent English word. We choose to use the American Sign Language Lexicon Video Dataset (ASLLVD) to

train numerous activity recognition architectures and assess their performance.

2. Related Work

There has been significant work done previously in the field of activity recognition for many different purposes. Oniga and Suto use a feed-forward backpropagation two-layer neural network architecture with 10 neurons on the hidden layer and 5 neurons on the output layer to classify different human body postures [8]. Using Levenberg-Marquardt as training algorithm and MSE for performance function evaluation, they achieved an accuracy of 99.96% when they attempted to classify five different body postures. They propose that this data can be used in the health field to detect unmonitored patients' postures and movements.

The task of gesture recognition has been studied using other classification methods that are not neural networks. Singha and Das used Karhunen-Loeve Transforms to achieve an accuracy of 96% in classifying one-handed gestures, but these were just from still images [10]. They pre-processed the images using edge detection and hand cropping to mitigate background noise, and then perform the transform, which creates a new coordinate system. Finally, they use two different linear classifiers to detect the final image. Both of these classifiers performed almost identically, one being based on the Euclidean distances of the images and another based on the angle of the Eigen vector with a reference axis. These gestures were not American Sign Language, but are very similar to how one would classify the ASL alphabet.

There has been a significant increase in the amount of work focusing on activity recognition in the years since 2014. This is largely due to the development of many public activity recognition datasets, and the popularization of neural network approaches. Karpathy et al presented various methods for utilizing convolutional neural nets in activity recognition tasks [5]. In particular Karpathy et al develop an architecture to expand CNN models into the time domain while maintaining the speed of such networks. Developing methods to integrate the temporal dimension of videos into the proven-successful CNN has been the focus of many activity recognition researchers recently [12] [4] [1]. The success of these methods continues to spur interest in improving action recognition capabilities and datasets.

The specific task of ASL recognition and classification has been done before with both neural networks and linear classifiers. Starner and Pentland used Hidden Markov Models for an accuracy of 99.2% [11]. They had the signer wear bright "tracking" gloves in each of the videos, and when they used no glove, they still achieved an accuracy of 92%. Hidden Markov Models can produce high accuracy, but they are required to be defined strictly before training, causing issues when dealing with a larger lexicon. Starner and Pentland used a forty word lexicon, but their research was a good foundation for later attempts at creating a comprehensive ASL classifier and translator.

Although there have been impressive results using linear models like Hidden Markov Models, neural networks have not been as thorough in classifying ASL hand-gestures with real-time video. The main advantage of neural networks compared to linear models is that they can identify the most salient features of the videos themselves; however, they require much larger datasets and take significantly longer to train. Mekala et al. were able to achieve 100% accuracy on identifying the American Sign Language alphabet using feature extraction in combination with a three-layer neural network architecture [7]. The features that they extract have to do with the finger and hand positions of the signs, specifically the locations of the tips of the fingers and the center of the palm. This accuracy is impressive, however, this is achieved for detecting individual characters through static signs, not dynamic signs in videos.

Even though the task of ASL recognition has been explored in these previous papers and more, the majority of them focus on classifying the twenty six letters, or classifying a small (¡20) number of gesture labels. This project is focused on attempting to classify 213 classes, which is a scale that has not been done before using a neural network architecture.

3. Dataset

We are using the American Sign Language Lexicon Video Dataset (ASLLVD) to train, validate, and test our models: `http://csr.bu.edu/asl/asllvd/annotate/index.html`, which contains almost 10,000 signing videos (1 - 10 videos per word). In the description of the dataset, it states that each video is the same resolution and same number of frames. This is actually not the case. Each video has a different resolution and number of frames, as well as different frames per second. To run this data through a model, all dimensions of each video need to be consistent across all videos. Therefore, we wrote a script that does that. The details of the script are discussed below. Additionally, we trimmed the dataset down due to limitations of our hardware's memory.

Figure 1. Example frame of one of the signers signing two different words.

The videos included in the dataset contain varied degrees of low-quality segmentation of the signer. These segmentations demonstrate a deviation from the conventional approach for activity recognition datasets. We hypothesize that these segmentations may help the model gain sensitivity to motion, rather than being dominated by spatial information; however, due to time-constraints, this hypothesis is not tested in our project.

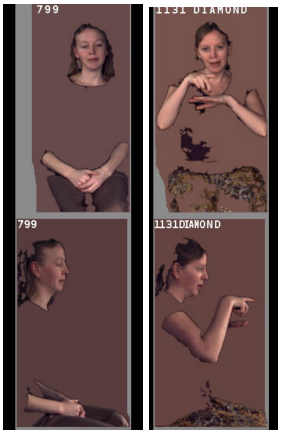


Figure 2. Examples frames showing the top and bottom half of a video. Left: Apple, Right: Diamond.

Each video of the dataset is divided in half, where the top half is of the signer facing toward the camera and the bottom half is of the signer turned 90 degrees. This is one major distinction that makes this dataset unique from previous works' datasets. In other attempts at classifying American Sign Language in videos, they have only used one camera angle, and therefore had difficulties due to lack of depth perception in the video. By having the bottom half from a different angle, this gives more detail to the data. Along with the elimination of background noise due to the green screen, we therefore predict that this dataset will prove to contribute to high accuracy ratings for our models.

4. Methods

4.1. Preprocessing

We had to perform many operations on the videos of the dataset in order to make sure all of the videos had the same resolution, number of frames, and frames per second. To start, we found the smallest height and width of each frame and smallest number of frames out of our entire dataset.

We noticed the beginning and ending of most, if not all, of the videos did not contain any useful information. The signers just sat there for a few seconds before starting to sign and ended with them stationary as well. Because of this, if we cut the frames of a video, we cut them from the front and the back. If a video had more frames than the smallest number of frames found, then we cut the frames of that video.

Next, we performed seam carving on the videos whose frames were larger than that of the video with the smallest resolution. We couldn't perform seam carving on each frame of a video individually.because it could be the case that the seam carving algorithm would remove different sections of each frame due to each frame containing a different energy map. Therefore, we created an energy map of the entire video and removed the seam that would cause the least significant change throughout the entire video. We performed this operation the number of times needed to reduce the size of each frame of a larger video to the size of the frames of the smallest video. The equation for energy map we used is defined below.

$$E = \sum_{f} Sobel(grayscale(image_f)) \tag{1}$$

Lastly, we process the data to only include classes which have at least 6 video samples. This decreases our dataset to include 1521 videos from 213 classes. This allowed us to train more accurately, while still achieving our project goals. We then broke these videos into frames at 5fps, leaving us 26560 images across our 213 classes.

4.2. Models

We implement five models in this project. For all of our models, we simplify the inputs by reading in frames from the video in sequences of 60 frames, sampled at 5 frames per second. Due to the varied nature of these models we train each for different number of epochs. Since the Inception model is a 2D-CNN and we used transfer learning, we were able to train for 20k epochs in short amounts of time. We trained both the LSTM and the LCRN for 200 epochs, the MLP model for 120 epochs, and the 3D-CNN for 50 epochs.

Our first model utilizes 2D-CNNs to classify videos based on individual frames. We classify each frame of a video independently, counting each frame as a vote for a class, then we select the class with the most votes as the class of the video. We will consider this model as a special case when evaluating our results due to it's use of an extremely accurate 2d-CNN. The 2D-CNN uses the InceptionV3 architecture and we retrain the final fully-connected layer using tensorflow. This model was used since it is exclusively reliant on spatial features of the videos for classification. We hypothesize that due to it's sole reliance on spatial features, this method will not be sufficient to achieve a high accuracy.

Our second model uses extracted CNN features again, but this time features are flattened and fed into an MLP network (fig 3). By using an MLP, we essentially leave it to the network to learn temporal features from the data. We

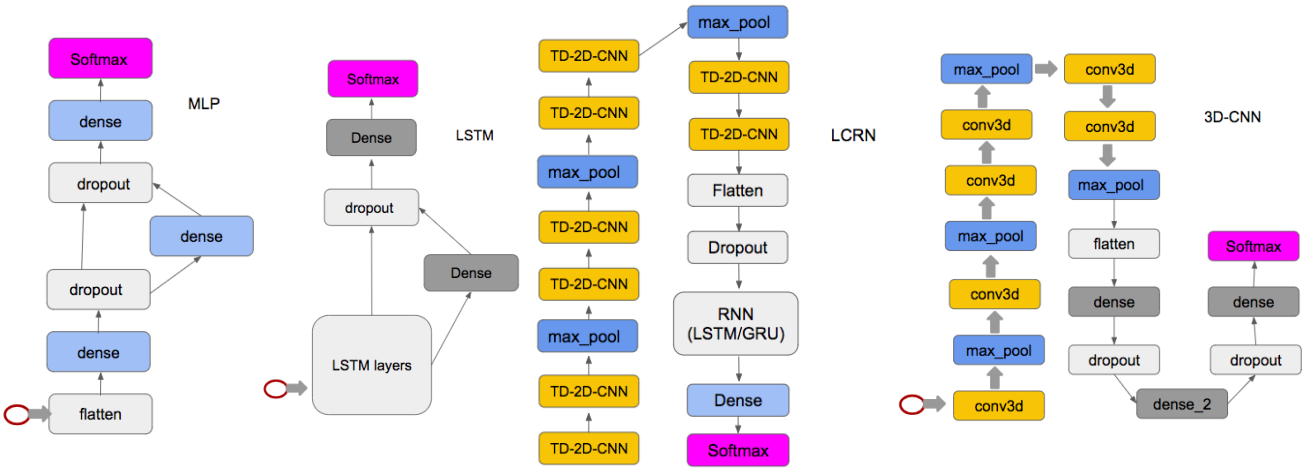


Figure 3. Architectures for our target models

use this approach as a stepping stone towards full utilization of the temporal dimension of the videos.

Our third model uses extracted features from each frame as the input to an RNN with LSTM layers (fig 3). The features are extracted from the pool layer of an InceptionV3 2D-CNN retrained for our dataset. We anticipate this LSTM model will have higher performance than the 2D-CNN approach, however, we anticipate the sub-optimal results due to our small dataset.

The fourth model we implement is a 3D-CNN based on the C3D Network proposed by Tran et al. [6] (fig 3). Due to the size of 3D-CNNs, we were forced to scale down the size of C3D network. However, we anticipate the network to have comparable performance to the RNN-LSTM network.

Lastly, our fifth model is a time-distributed CNN feeding into an RNN to make predictions (fig 3). This type of model mirrors that of a Long-term Recurrent Convolutional Network (LRCN) [2]. In this model, we experiment we use the VGG-16 model as a starting point for our CNN structure, then we experiment with the RNN cell types and structures, using both LSTM and GRU cells. We anticipate that using a GRU cell type with this model will produce the greatest accuracy due to the tendency for GRUs to be better suited for use when data is sparse.

5. Results

5.1. Dataset

Due to hardware limitations, we were forced to cut down the original dataset by a large margin. We chose to only use sign language labels that had more than six videos associated with them. As a result, our final dataset size was 1521 videos,consisting of 213 classes. We split this data into a training, validation, and test set with the following rules: both validation and test sets contained one video from each label, while the training set contained the rest, or in other words, $n_i - 2$ videos per label i, where n_i is the number of videos in label i.

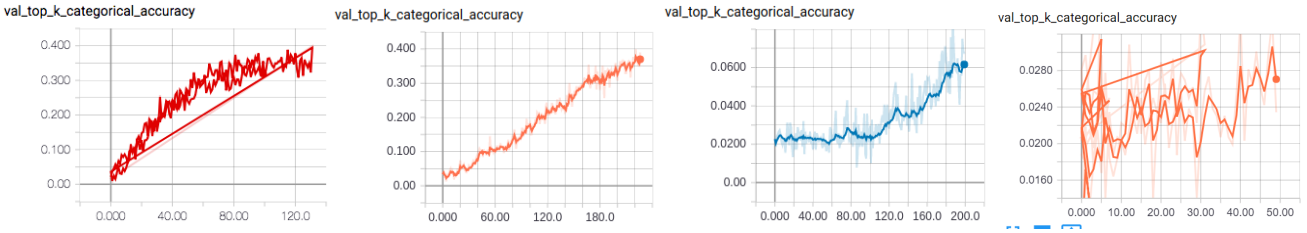
Model	Accuracy
Baseline	0.0046
InceptionV3	0.840
MLP	0.181
LSTM	0.168
3D-CNN	0.010
LRCN	0.012

Figure 4. testing accuracy from all models

5.2. Evaluation

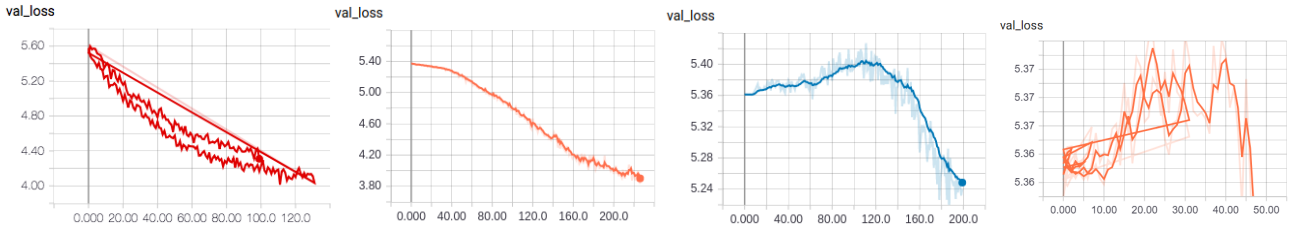
We arrived at our final architectures through hyperparameter tuning and structure testing. In particular, we experimented with using GRU cells rather than LSTM cells in our LRCN model, and we altered the depth and width of the MLP and 3D-CNN models. The MLP model consistently overfit, while the 3D-CNN was consistently unable to train in a reasonable number of epochs.

We present the results of the fine-tuned models being run on the test set in figure 4. From this, we see that the Inception model achieved the highest accuracy at 84%. However, the remaining models performed relatively poorly in comparison, not even reaching 20% accuracy. These discrepancies are troubling, yet this seems to make sense when considering the Inception video classification pipeline described above. If we were attempting to classify individual video frames into one of 213 classes by choosing randomly, we'd have a $\frac{1}{213}$ probability of selecting correctly, as defined by our baseline model. Given this probability, if we then classify 60 frames independently, then we have a $\frac{1}{213}\dot{6}0 = .276$ probability of guessing the correct class.



(a) MLP (b) LSTM (c) LRCN (d) 3D-CNN

Figure 5. Top-5 Categorization accuracy for our 4 temporal models on Validation set.



(a) MLP (b) LSTM (c) LRCN (d) 3D-CNN

Figure 6. Loss plots for our 4 temporal models on Validation set.

Thus, the baseline comparison for the Inception net classification methodology is far higher than for our other models. The test set accuracy is still high in comparison however. This can be attributed to the InceptionV3 architecture itself; the model was able to be retrained very accurately. Looking at the training and validation loss and entropy gives us a better sense of the capabilities of the retrained Inception net:

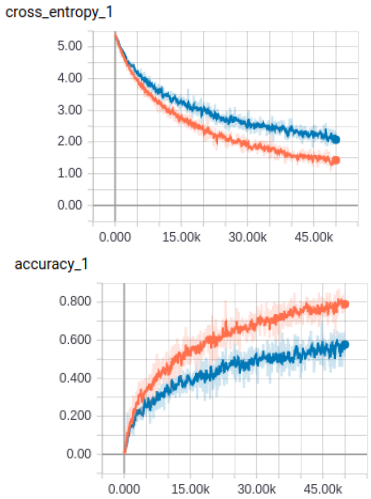


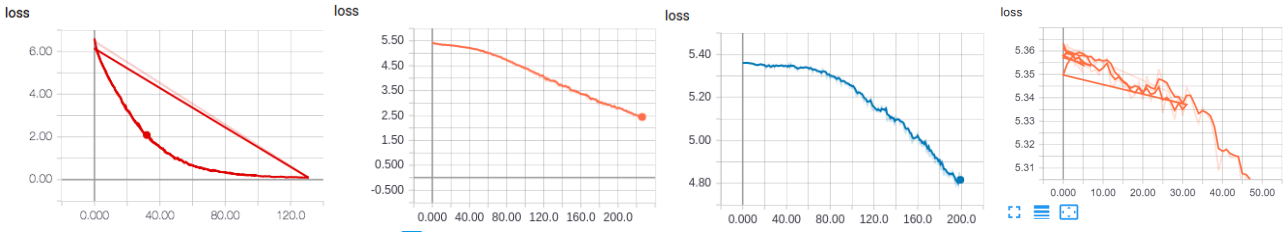
Figure 7. InceptionV3 loss (top) and accuracy (bottom) from all models. Training is in orange, validation is in blue.

From this, we can see that even after increasing the training epochs to 50k, there appears to still be room for improvement as the entropy has not completely flattened out. Thus, our hypothesis that the Inception model would have poor performance proved to be false. On our data in particular, the Inception net seemed extremely capable of differentiating between many classes, achieving over 50% validation accuracy in classifying an individual image (fig. 6). We see the training and validation accuracy begin to deviate around 15k epochs. This is most likely due to the redundancy of the data. There are only 15 signers throughout the entire dataset, and some signs appear similar when individual frames are considered. Thus, by ignoring the temporal component of the videos, we introduce redundancy into the model, which decreases the model ability to generalize.

We now move on to evaluating our remaining models. From the model accuracies (fig. 4), we see that these models did not perform nearly as well in comparison to the Inception model. We can immediately see that the 3D-CNN and the LRCN models performed poorly in comparison to the MLP and LSTM models. Both the 3D-CNN and the LRCN did only slightly better than our baseline random model. We can see from the top-5 categorization accuracy plots (fig. 5) that the top-5 accuracy was beginning to increase, while the loss began to rapidly decrease (fig 6). For the LRCN in particular, the accuracy was beginning to increase much faster after epoch 120. Looking at the validation loss for the LRCN model (fig. 6c) further supports this point.

In looking at the architecture for both the 3D-CNN and the LCRN, we note the presence of many more layers and convolution layers in these models than in the MLP and LSTM models. These deeper networks, paired with the convolution layers in particular most likely contributed to the slow accuracy improvement for these networks.

Now looking at the MLP model, we see that the model



(a) MLP (b) LSTM (c) LRCN (d) 3D-CNN

Figure 8. Loss plots for our 4 temporal models on Validation set.

clearly overfit our training data since training loss quickly goes to 0. This might be attributable to the CNN features used for training the MLP model appearing similar to one another due to the redundancy of our data. We were unable to tune and alter the MLP model to a point that fit the data well without overfitting, and thus the accuracy of an MLP model is limited to about 20%. Turning now towards the LSTM model, we can see from both the training and validation losses(fig 6, 8), that we have underfit our data, and would have benefited from further training. Though these results fail to compare to the Inception model, we can see that the LSTM model in particular is able to quickly learn to classify videos based on features from their frames.

6. Conclusion

In conclusion, we've presented 5 models to be used in activity recognition over videos. We find that the Inception approach proves accurate, however we presume such an approach would not generalize well on this dataset. Due to the segmentation and formatting of the videos in the ASLLVD, the Inception 2D-CNN is most likely matching the format of the frames, and is not learning the true motion of the signs. Furthermore, the MLP model proves to be too simple to learn to distinguish the temporal features of the grouped frames. The LSTM, LRCN, and 3D-CNN appear to be most reliable for tackling such tasks, however LSTM models are able to learn to fit the data significantly faster than LRCN and 3D-CNN. In relation to the task of constructing an ASL automated translator, the model architectures appear to be developed to an acceptable state for this purpose, however the ASLLVD does not provide a reasonable data source for training activity recognition models. The data is inconsistent and sparse. Thus, a large, consistent, and processed dataset is needed to in order for such a translator to be constructed in the future. Due to time constraints, we used simplified models, smaller sequence sizes, and naive feature representations. In the future, it would be beneficial to expand the breadth of such a review to include more complex models that utilize two-stream approaches and optical flow images. Additionally, having more resources available to train larger models for longer periods of time will serve to results.

References

[1] Y. Chao, S. Vijayanarasimhan, B. Seybold, D. Ross, J. Deng, and S. R. Rethinking the faster r-cnn architecture for temporal action localization. 2018.

[2] H. L. R. M. V. S. G. S. S. K. Donahue, J. and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. 2016.

[3] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. 2016.

[4] R. Hou, C. Chen, and M. Shah. Tube convolutional neural network (t-cnn) for action detection in videos. 2017.

[5] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. 2014.

[6] C. Ma, M. Chen, Z. Kira, and G. AlRegib. TS-LSTM and Temporal-Inception: Exploiting Spatiotemporal Dynamics for Activity Recognition. *ArXiv e-prints*, 1703(10667), 2017.

[7] P. Mekala, Y. Gao, and J. Fan. Real-time sign language recognition based on neural network architecture. 2011.

[8] S. Oniga and J. Suto. Human activity recognition using neural networks. *ICCC*, 2014.

[9] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. *In Advances in Neural Information Processing Systems*, pages 568–576, 2014.

[10] J. Singha and K. Das. Hand gesture recognition based on karhunen-loeve transform. 2013.

[11] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. 1997.

[12] H. Zhu, R. Vial, and S. Lu. Tornado: A spatio-temporal convolutional regression network for video action proposal. 2017.