2021

# Contextual Understanding of Sequential Data Across Multiple Modalities

Sangwoo Cho
*University of Central Florida*

# CONTEXTUAL UNDERSTANDING OF SEQUENTIAL DATA ACROSS MULTIPLE MODALITIES

by

SANGWOO CHO
M.S. University of North Carolina at Chapel Hill, 2014
M.E. Korea University, 2007
B.E. Sogang University, 2005

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2021

Major Professor: Hassan Foroosh

# ABSTRACT

In recent years, progress in computing and networking has made it possible to collect large volumes of data for various different applications in data mining and data analytics using machine learning methods. Data may come from different sources and in different shapes and forms depending on their inherent nature and the acquisition process. In this dissertation, we focus specifically on sequential data, which have been exponentially growing in recent years on platforms such as YouTube, social media, news agency sites, and other platforms. An important characteristic of sequential data is the inherent causal structure with latent patterns that can be discovered and learned from samples of the dataset. With this in mind, we target problems in two different domains of Computer Vision and Natural Language Processing that deal with sequential data and share the common characteristics of such data. The first one is action recognition based on video data, which is a fundamental problem in computer vision. This problem aims to find generalized patterns from videos to recognize or predict human actions. A video contains two important sets of information, i.e. appearance and motion. These information are complementary, and therefore an accurate recognition or prediction of activities or actions in video data depend significantly on our ability to extract them both. However, effective extraction of these information is a non-trivial task due to several challenges, such as viewpoint changes, camera motions, and scale variations, to name a few. It is thus crucial to design effective and generalized representations of video data

that learn these variations and/or are invariant to such variations. We propose different models that learn and extract spatio-temporal correlations from video frames by using deep networks that overcome these challenges. The second problem that we study in this dissertation in the context of sequential data analysis is text summarization in multi-document processing. Sentences consist of sequence of words that imply context. The summarization task requires learning and understanding the contextual information from each sentence in order to determine which subset of sentences forms the best representative of a given article. With the progress made by deep learning, better representations of words have been achieved, leading in turn to better contextual representations of sentences. We propose summarization methods that combine mathematical optimization, Determinantal Point Processes (DPPs), and deep learning models that outperform the state of the art in multi-document text summarization.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1
# INTRODUCTION

Ever since the internet and handheld devices have become ubiquitous, huge number of sequential data have been generated and are being produced on daily basis by people and machines: People take millions of videos of their families, friends, and gatherings to share on the internet around the world; News media create daily articles and followup stories about new events; and output of sensors such as webcams and surveillance cameras deployed for various applications, such as security, weather monitoring, and manufacturing are continuously shared on the internet, and often in real time. People frequently use or interact with various types of sequential data such as videos, news, and emails on daily basis. It is therefore of paramount interest to not only understand and make sense of such data over time, but also to distill and summarize the overwhelmingly growing data for more efficient human consumption, while preserving integrity and faithfulness to the original content and intent. The sequential data usually contain latent patterns that reflect their essential underlying information. With the rise of deep neural networks [117, 210, 211], one can effectively extract those important information with a data-driven training-based approach. In this dissertation, we study the special nature of sequential data by investigating two different problems that share the aforementioned characteristics of sequential data, i.e. video-based human action recognition, and text summarization. Each problem takes as input some sequential data: a video

is a collection of sequential frames; and text is a sequence of sentences, while a sentence is an ordered group of words. To emphasize the shared nature of such sequential data, we show that videos of human actions can be modeled in an abstract way in terms of what we refer to as "action words" and "action sentences".

We recognize that one common thread in better understanding of sequential data (regardless of their superficial differences, e.g. videos versus text), is the effective extraction of their hidden representations or contextual information. Therefore, we make the following important contributions in terms of the methods and results that are designed to extract such contextual information in different problems and different domains.

- We propose neural network models that extract temporal correlation information of different modalities in human action recognition. Temporal Convolutional Neural Networks (CNNs) with various sizes of kernels are proposed to extract different local hidden patterns in order to distinguish the semantically dissimilar human actions, effectively.

- To obtain spatio-temporal context information, we propose a novel approach to merge temporal changes of appearance and motion data. The proposed network fuses the temporal associations of appearances and motions leading to acquiring video-level context.

- In order to understand core hidden patterns of human actions, skeleton-based information is often used. Unlike the CNN or recurrent neural networks that focus only on local relations, self-attention networks consider all possible pairwise associations in temporal order. We

propose Self Attention Networks that effectively obtains temporal correlations to understand human movements based on skeleton information.

- There are many obstacles to achieve a quality multi-document summarization system since the document contains excessive redundant information and text understanding is challenging. We present a system exploiting capsule networks for extracting context between pair sentences. We model pairwise sentence similarity as Determinantal Point Processes (DPP) that choose a set of summary sentences that are both representative and attain diversity.

- Language models are trained with a huge number of text data and contain abundant context information that can be used in other tasks. To improve the similarity measure and the importance measure of sentences, we propose a model based on Bidirectional Encoder Representation from Transformers (BERT) to measure both similarity and importance effectively. By combining DPP with the contextualized representations, we achieve summarization results that outperform the state of the art.

- Amongst the best means to summarize text (which we believe could inspire video summarization) is *highlighting*. We propose a method of generating summary highlights, overlaid on the original documents, to make it easier for readers to sift through a large body of text. The method allows for the summaries to be understood in context in order to prevent the summarizer from distorting the original intent and meaning, a problem that abstractive summarization methods are known for. In particular, we present a new method to produce *self-contained* highlights that are understandable on their own to avoid any confusion.

## 1.1 Temporal Context for Action Recognition and Prediction

Video-based action recognition is an active research area due to its important practical applications in many areas, such as video surveillance, behavior analysis, and human-computer interaction. The action recognition task is accomplished after acquiring the entire video, while action prediction is different in the sense that it aims at classifying the action with shortest possible latency, i.e. classify as early as possible as the frames come in. The capability of predicting an action early is crucial in both surveillance systems and human-computer interaction. The two tasks of action prediction and recognition have often been researched separately under different settings and constraints.

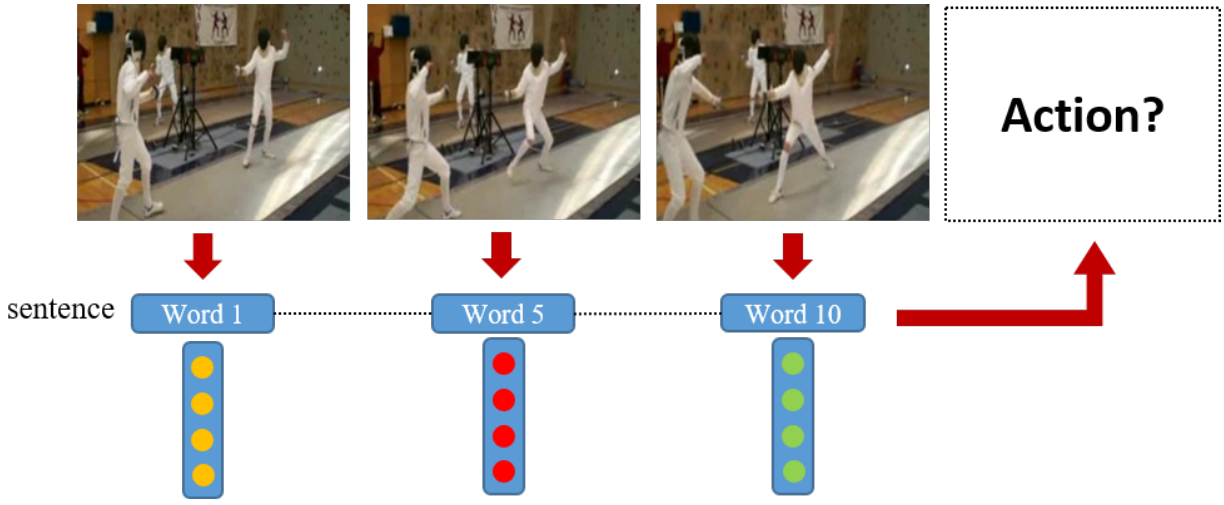



Figure 1.1: Given a partial or a full video frames, our goal is to classify the correct action. Each frame is converted to a corresponding "action word" and the sequence of "action words" is trained to predict an activity.

A video contains two important pieces of information: appearance and motion. These information are complementary, and therefore an accurate prediction relies on the ability to extract the information with low latency, i.e. as early as possible in the temporal sequence. However, extracting effective information (whether for prediction or recognition) is non-trivial, due to a number of difficulties such as viewpoint changes, camera motions, and scale variations, to name a few. It is thus crucial to design an effective and generalized representation of a video. Convolutaional Neural Networks (ConvNets) [96] have been playing a key role in solving hard problems in various areas of computer vision, e.g. image classification [96, 71, 221] and human face recognition [151]. ConvNets also have been employed to solve the problem of action recognition [164, 83, 191, 134, 212] in recent literature.

Data-driven supervised learning enables to achieve discriminating power and proper representation of a video from raw data. However, ConvNets for action recognition have not shown a significant performance gain over the methods utilizing hand-crafted features [200, 141] or feature-independent methods [175]. We speculate that the main reason for the lack of big impact is that ConvNets employed in action recognition do not take full advantage of temporal sequencing or order. Recently some methods [195, 36] attempted to capture long-term temporal information. However, they require excessive computation for a long video.

Inspired by key ideas from Natural Language Processing (NLP), and as a contribution in this dissertation for modeling temporal context, we represent each frame as a word and a video as a sequence of such words. The sequence of words, or a sentence, is a new video representation as shown in Fig. 1.1. We call this abstract representation an *Action Word*. We use the standard Bag of

Words (BoW) [140] framework to encode each visual feature as an assigned word in a codebook. The sequence of words then is learned with a simple but effective CNN architecture capturing the sequential order of temporal information. This method is flexible to input size, and hence is applicable to any length of videos. The capability to adopt a variable-size input, combined with low latency versus high accuracy makes the method particularly powerful for both action prediction and action recognition.

Our key contributions can thus be summarized as follows: (i) A new representation for video data as a sequence of words that inherently captures temporal order and sequencing of information. (ii) An effective ConvNet that learns such temporal sequencing to predict with low latency an action. (iii) The ability of the method to maintain state-of-the-art accuracy in both prediction and recognition with the challenging datasets, such as UCF101 and HMDB51. (iv) The entire system is easy to implement and is trained with a small computational cost compared to other methods employing ConvNets.

## 1.2 Spatio-Temporal Fusion Networks for Action Recognition

Video-based action recognition is an active research topic due to its important practical applications in many areas, such as video surveillance, behavior analysis, and human-computer interaction. Unlike a single image that contains only spatial information, a video provides additional motion information as an important cue for recognition. Although a video provides more information, it is non-trivial to extract the information due to a number of difficulties such as viewpoint changes,

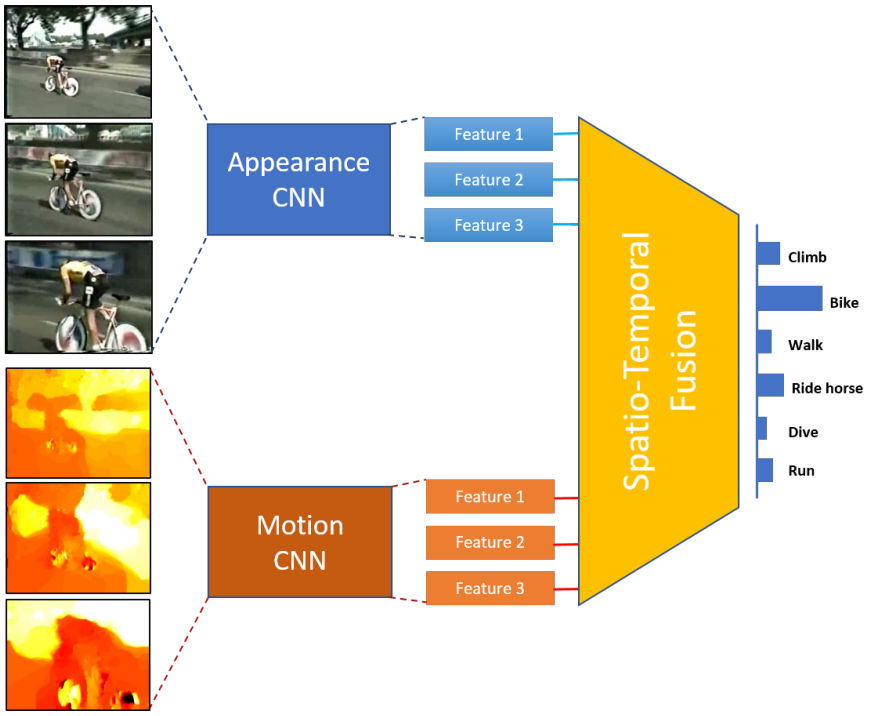



Figure 1.2: An illustration of spatio-temporal fusion network (STFN) for action recognition. Given multiple segments of a video, the network extracts temporal dynamics of appearance and motion cues and fuses them to build a spatio-temporal video representation via end-to-end learning. The appearance and motion ConvNets share the same weights and are employed to extract appearance and motion features, respectively.

camera motions, and scale variations, to name a few. It is thus crucial to design an effective and generalized representations of a video.

In recent years, two-stream ConvNets [164] have become popular in action recognition, attempting to exploit both the appearance and motion data. This, in a sense, is also aiming to increase the performance gain by ConvNets over hand-crafted features [200, 141], as pointed out

earlier. However, the two data streams are typically trained with separate ConvNets and only combined by averaging the prediction scores. This approach is not helpful when the two information are needed simultaneously, e.g. motions of brushing teeth and brushing hair are similar, and therefore appearance information is needed to discriminate them. Due to the lack of spatio-temporal features for action recognition, several methods [192, 46, 26] have attempted to incorporate both sources of information. They typically take frame-level features and integrate them using an RNN [71] network and temporal feature pooling [47, 132, 206] in order to incorporate temporal information. However, they still lack in extracting a representation that captures video-wide temporal information.

As part of this dissertation, we investigate a proper model to fuse the appearance and motion dynamics to learn a video level spatio-temopral representation. The proposed spatio-Temporal Fusion Network (STFN) aggregates different size of local temporal dynamics in multiple video segments and combines them to obtain a video level spatio-temporal representation. STFN is mainly motivated by two components: a residual-inception module [134], and 1D convolution layers [106]. The former is suitable for extracting latent features and the latter works well in extracting temporal dynamics. We modified the original residual-inception module [134] and designed a new block for spatio-temporal fusion that achieves our research goals. The new residual-inception block processes local and global temporal dynamics for each data. Given the extracted dynamic information, appearance and motion dynamics are merged with fusion operations for spatio-temporal features. This method overcomes the previous drawback, i.e. the lack of utilizing video-wide temporal infor-

mation, and learning spatio-temporal features. We investigate a variety of different fusion methods and perform ablation studies to find the best network.

Our key contributions in this part of the dissertation can thus be summarized as follows: (i) A convolution block, effective to extract temporal representations, is proposed. (ii) A novel ConvNet is introduced to learn spatio-temporal features effectively by fusing two different features properly. (iii) the proposed STFN achieves state-of-the-art performance on the two challenging datasets, UCF101 (95.4%) and HMDB51 (72.1%). (iv) The entire system is easy to implement and is trained by an end-to-end learning of deep networks.

### 1.3 Skeleton-Based Action Recognition with Self-Attention Networks

Video-based action recognition has been an active research topic due to its important practical applications in many areas, such as video surveillance, behavior analysis, and video retrieval. Human action recognition can also be applicable to human-computer interaction or human-robot interaction to help machines understand human behaviors better [218, 145, 22]. Unlike a single image that contains only spatial information, a video provides additional motion information as an important cue for recognition. Although a video provides more information, it is non-trivial to extract the information due to a number of difficulties such as viewpoint changes, camera motions, and scale variations, to name a few. There has been extensive research in RGB video-based action recognition and one of the mainstream methods is to employ both temporal optical flow and spatial appearance to obtain spatial and temporal information [165] . The RGB video datasets typically

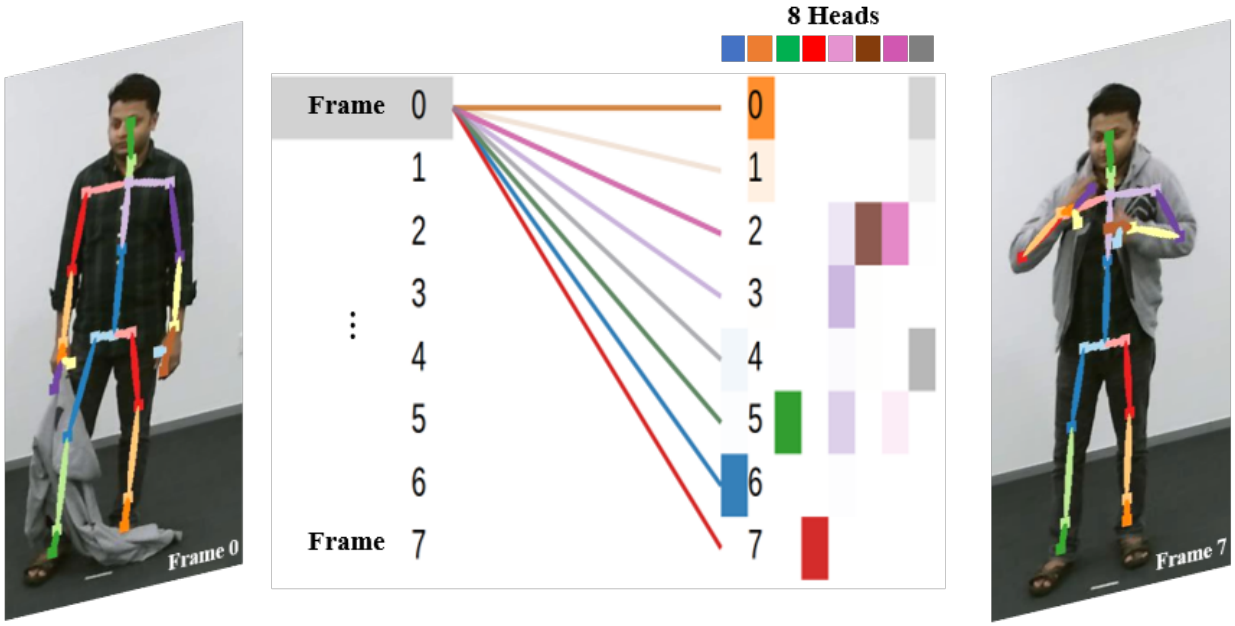



Figure 1.3: An example of self-attention response from the last self-attention layer. Eight frames are uniformly sampled from an action with the class 'put on jacket' and illustrated as frame 0 to 7. Frame 0 has the strongest correlation with the last frame, frame 7, at the fourth head ▬, and attends heavily itself at the second head ▬. Note that with the self-attention network each frame is associated with other frames so that local and global context information can be acquired.

contain an extensive amount of data to process, hence require large models and resources to train them properly. On the other hand, skeleton based action recognition comprises of only key joint locations of human bodies [4, 158, 160, 161, 173, 174]. With the advent of cost-effective depth cameras [232], stereo cameras, and the advanced techniques for human pose estimation [11, 159], the cost to obtain key points has reduced. As a result, skeleton-based human action recognition has regained and garnered increasing attraction in recent years [2, 39, 222]. Although, key joint

locations do not include appearance information, humans are able to recognizing actions from the motion of a few human skeleton joints according to Johansson [78]. In this part of the dissertation, we further study temporal context in human action recognition, when focusing solely on 3D skeleton sequences.

To extract information from skeleton sequences, many works naturally apply recurrent neural networks (RNNs) to model the temporal dynamics [154, 118, 231]. They also utilize CNNs to model spatio-temporal dynamics by treating the 3D skeleton data as 2D pseudo images with 3 channels [110, 213]. Another method is to retrieve structure information of human body by constructing a graph with human joints as edges [222], which is also based on CNNs. Despite significant progress and improvements in performance, the problem as a whole and many aspects of it are still considered as not fully solved. Both recurrent and convolutional operations are neighborhood-based local operations [216] either in space or time; hence local-range information is repeatedly extracted and propagated to capture long-range dependencies. Many works have designed networks with hierarchical structures [39, 109, 21] to obtain longer range and deeper semantic information, but the problem still persists if there are back and forth semantic dependencies.

In this dissertation, we propose a novel model based on a Self-Attention Network (SAN) to overcome the above limitation and retrieve better semantic information (Fig. 1.3). Fig. 1.4 shows the overall pipeline of our model. The framework is motivated by temporal segment network [207] that extracts short-term information from each video sequence. Our model extracts semantic information from each video sequence by SAN variants. SAN-Variants take a sequence of features

from encoded signals and compute the response at each position as a weighted sum of features at all positions. This operation enables SAN-Variants to correlate features in distance or even in opposite directions. The predicted outputs based on each clip are merged with consensus operations to capture deeper semantic understanding. Therefore, our model can effectively solve the problem of acquiring long-term semantic information. Experimental results show that the learned SAN variants outperform state of the art methods on challenging large scale datasets. We also visualize the attention correlations trying to understand how the network works and provide some insights. The main contributions of the dissertation here are summarized as follows:

1. We propose Self Attention Network (SAN) variants SAN-V1, SAN-V2 and SAN-V3 for effectively capturing deep semantic correlations from action sequences involving human skeleton.
2. We have integrated the Temporal Segment Network (TSN) with our SAN variants. We observed improved performance because of this integration of TSN and SAN variants.
3. We visualize self-attention probabilities to show how each frame is correlated with other frames.
4. Our proposed method achieves state-of-the-art results on two large scale datasets: NTU RGB+D and Kinetics-skeleton

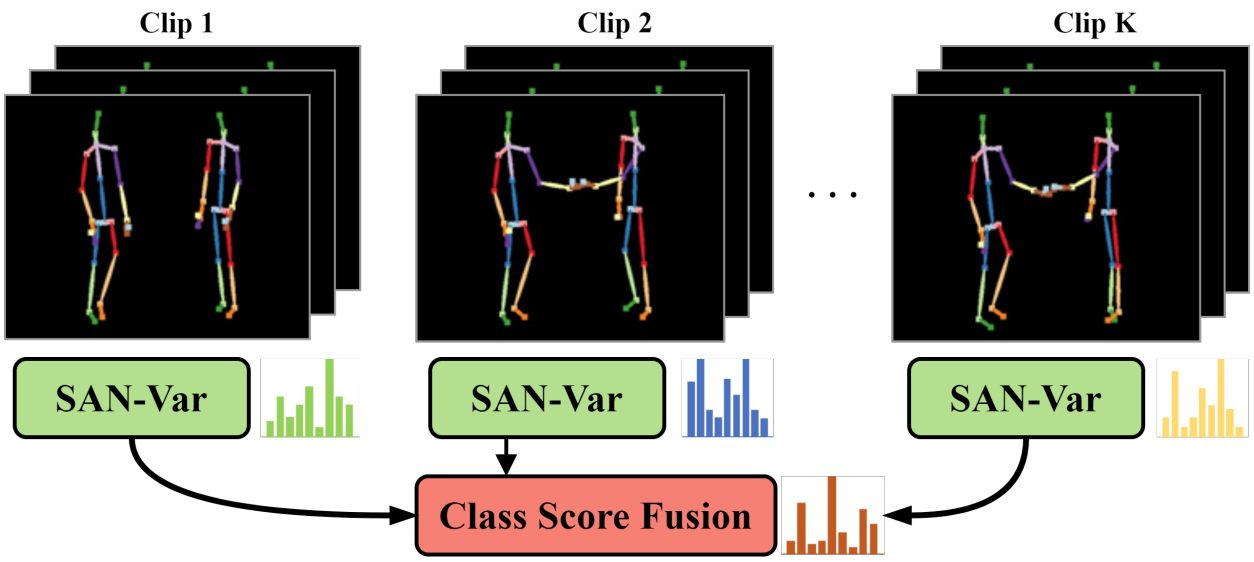



Figure 1.4: The overall pipeline of the proposed model. The network takes as inputs temporally segmented clips and extracts contextual information from each snippet by one of SAN variants described in section 5.2.3. Predictions of each snippet are fused to compute the final prediction.

## 1.4 Text Summarization with Determinantal Point Processes and Capsule Networks

Multi-document summarization is arguably one of the most important tools for information aggregation. It seeks to produce a succinct summary from a collection of textual documents created by multiple authors concerning a single topic [131]. The summarization technique has seen growing interest in a broad spectrum of domains that include summarizing product reviews [57, 223], student survey responses [122, 123], forum discussion threads [35, 187], and news articles about a particular event [72]. Despite the empirical success, most of the datasets remain small, and the cost of hiring human annotators to create ground-truth summaries for multi-document inputs can be prohibitive.

Impressive progress has been made on neural abstractive summarization using encoder-decoder models [147, 152, 139, 17]. These models, nonetheless, are data-hungry and learn poorly from small datasets, as is often the case with multi-document summarization. To date, studies have primarily focused on single-document summarization [152, 14, 98] and sentence summarization [128, 236, 12, 168] in part because parallel training data are abundant and they can be conveniently acquired from the Web. Further, a notable issue with abstractive summarization is the reliability. These models are equipped with the capability of generating new words not present in the source. With greater freedom of lexical choices, the system summaries can contain inaccurate factual details and falsified content that prevent them from staying "true-to-original."

In this dissertation, we instead focus on an extractive method exploiting the Determinantal Point process (DPP; Kulesza and Taskar, 2012) for multi-document summarization. DPP can be trained on small data, and because extractive summaries are free from manipulation, they largely remain true to the original. DPP selects a set of most representative sentences from the given source documents to form a summary, while maintaining high diversity among summary sentences. It is one of a family of optimization-based summarization methods that performed strongest in previous summarization competitions [58, 115, 100].

Diversity is an integral part of the DPP model. It is modelled by *pairwise repulsion* between sentences. In this dissertation, we exploit the capsule networks [69] to measure pairwise sentence (dis)similarity, then leverage DPP to obtain a set of diverse summary sentences. Traditionally, the DPP method computes similarity scores based on the bag-of-words representation of sentences [100] and with kernel methods [62]. These methods, however, are incapable of cap-

turing lexical and syntactic variations in the sentences (e.g., paraphrases), which are ubiquitous in multi-document summarization data as the source documents are created by multiple authors with distinct writing styles. We hypothesize that the recently proposed capsule networks, which learn high-level representations based on the orientational and spatial relationships of low-level components, can be a suitable supplement to model pairwise sentence similarity.

Importantly, we argue that predicting sentence similarity within the context of summarization has its uniqueness. It estimates if two sentences contain redundant information based on both surface word form and their underlying semantics. As an example, the two sentences "*Snowstorm slams eastern US on Friday*" and "*A strong wintry storm was dumping snow in eastern US after creating traffic havoc that claimed at least eight lives*" are considered similar because they carry redundant information and cannot both be included in the summary. These sentences are by no means semantically equivalent, nor do they exhibit a clear entailment relationship. The task thus should be distinguished from similar tasks such as predicting natural language inference [9, 219] or semantic textual similarity [15]. In this work, we describe a novel method to collect a large amount of sentence pairs that are deemed similar for summarization purpose. We contrast this new dataset with those used for textual entailment for modeling sentence similarity and demonstrate its effectiveness on discriminating sentences and generating diverse summaries. The contributions of this work can be summarized as follows:

- we present a novel method inspired by the determinantal point process for multi-document summarization. The method includes a *diversity* measure assessing the redundancy between sen-

tences, and a *quality* measure that indicates the importance of sentences. DPP extracts a set of summary sentences that are both representative of the document set and remain diverse;

- we present the first study exploiting capsule networks for determining sentence similarity for summarization purpose. It is important to recognize that summarization places particular emphasis on measuring redundancy between sentences; and this notion of similarity is different from that of entailment and semantic textual similarity (STS);
- our findings suggest that effectively modeling pairwise sentence similarity is crucial for increasing summary diversity and boosting summarization performance. Our DPP system with improved similarity measure performs competitively, outperforming strong summarization baselines on benchmark datasets.

### 1.5 Text Summarization with DPP and Contextualized Representations

Determinantal point processes (DPP) are one of a number of optimization techniques that perform remarkably well in summarization competitions [72]. These optimization-based summarization methods include integer linear programming [58], minimum dominating set [157], maximizing submodular functions under a budget constraint [115, 227], and DPP [101]. DPP is appealing to extractive summarization, since not only has it demonstrated promising performance on summarizing text/video content [62, 230, 156], but it has the potential of being combined with deep neural networks for better representation and selection [54].

The most distinctive characteristic of DPP is its decomposition into the *quality* and *diversity* measures [101]. A *quality* measure is a positive number indicating how important a sentence is to the extractive summary. A *diversity* measure compares a pair of sentences for redundancy. If a sentence is of high quality, any *set* containing it will have a high probability score. If two sentences contain redundant information, they cannot both be included in the summary, thus any *set* containing both of them will have a low probability. DPP focuses on selecting the most probable *set* of sentences to form a summary according to sentence quality and diversity measures.

To better measure quality and diversity aspects, we draw on deep contextualized representations. A number of models have been proposed recently, including ELMo [143], BERT [31], XLNet [224, 27], RoBERTa [121] and many others. These representations encode a given text into a vector based on left and right context. With carefully designed objectives and billions of words used for pretraining, they have achieved astonishing results in several tasks including predicting entailment relationship, semantic textual similarity, and question answering. We are particularly interested in leveraging BERT for better sentence quality and diversity estimates.

This dissertation extends on previous work [23] by incorporating deep contextualized representations into DPP, with an emphasis on better sentence selection for extractive multi-document summarization. The major research contributions of this work include the following: (*i*) we make a first attempt to combine DPP with BERT representations to measure sentence quality and diversity and report encouraging results on benchmark summarization datasets; (*ii*) our findings suggest that it is best to model sentence *quality*, i.e., how important a sentence is to the summary, by combining semantic representations and surface indicators of the sentence, whereas pairwise sen-

tence *dissimilarity* can be determined by semantic representations only; (*iii*) our analysis reveals that combining contextualized representations with surface features (e.g., sentence length, position, centrality, etc) remains necessary, as deep representations, albeit powerful, may not capture domain-specific semantics/knowledge such as word frequency.

### 1.6 Text Summarization with DPP and Sub-Sentence Highlights

A summary is reliable only if it is true-to-original. Abstractive summarizers are considered to be less reliable despite their impressive performance on benchmark datasets, because they can hallucinate facts and struggle to keep the original meanings intact [153, 97]. In this dissertation, we seek to generate summary highlights to be overlaid on the original documents to allow summaries to be understood in context and avoid misdirecting readers to false conclusions. This is especially important in areas involving legislation, political speeches, public policies, social media, and more [150, 95]. Highlighting is most commonly used in education to make important information stand out and bring attention of readers to the essential topics [146].

The characteristics of summary highlights are: *saliency*, i.e., highlights must give the main points of the documents; and *non-redundancy*, suggesting that redundant content cannot be repeated in a summary [131]. Importantly, a highlighted text should be *self-contained*, i.e., understandable on its own, without the need for specific information from surrounding context. Table 1.1 provides an example of sub-sentence highlights. As an example, "*New Jersey is located in*" hardly constitutes a good highlight because the information it contains is incomplete and may confuse

Table 1.1: An example of sub-sentence highlights overlaid on the original document; the highlights are self-contained.

**Original Document and Summary Highlights**

Afghan opium kills 100,000 people every year worldwide – more than any other drug – and the opiate heroin kills five times as many people in NATO countries each year than the eight-year total of NATO troops killed in Afghan combat, *the United Nations said Wednesday.* About 15 million people around the world use heroin, opium or morphine, fueling a $65 billion market for the drug and also fueling terrorism and insurgencies... *Drug money is funding insurgencies* in Central Asia, which has huge energy reserves, Costa said... Europe and Russia together consume just under half of the heroin coming out of Afghanistan, the United Nations concluded, and *Iran is by far the single largest consumer of Afghan opium.*

readers. To date, there has not been any unified framework to account for all these characteristics to generate highlights. We overcome the challenge by identifying self-contained sub-sentence segments from documents, then combining determinantal point processes and deep contextualized representations to produce highlights.

Determinantal point process belongs to a class of optimization methods that have had considerable success in summarizing text and video [101, 62, 156]. It selects a diverse subset from a ground set of items, where an item is a candidate text segment in the context of generating summary highlights. An item is characterized by a *quality* score that indicates the salience of the segment and a *diversity* score that models *pairwise repulsion*, suggesting that two segments carrying similar

meaning cannot both be included in the summary to avoid redundancy. The quality and diversity decomposition of DPP allows it to identify an optimal subset from a collection of candidate segments.

We study sub-sentence segments as they strike a balance between the quality and amount of highlights. Whole sentences often contain excessive or unwanted details; keywords are succinct but less informative. We conjecture that sub-sentence segments can be identified from a document similar to salient objects are identified from an image using bounding boxes [61]. To best estimate the size of segments, we present a novel method to "overgenerate" a rich set of self-contained, partially-overlapping sub-sentence segments from any sentence based on contextualized representations [225, 33], then leverage determinantal point processes to identify an essential subset based on saliency and non-redundancy criteria. Our contributions of this work are summarized as follows.

- We propose to generate sub-sentence summary highlights to be overlaid on source documents to enable users to quickly navigate through content. Comparing to keywords or whole sentences, sub-sentence segments allow us to attain a good balance between quality and amount of highlights.
- Importantly, sub-sentence segments are designed to be self-contained, and for which we introduce a new algorithm based on deep contextual representations to obtain self-contained text segments. All candidate segments are fed to determinantal point processes to identify an optimal subset containing informative, non-redundant, and self-contained sub-sentence highlights.

- We perform experiments on benchmark summarization datasets to demonstrate the flexibility and modeling power of our approach. Our analysis provides further evidence that highlighting offers a promising avenue of research.[1]

## 1.7 Dissertation Organization

The rest of the dissertation is organized as follows. In Chapter 2, we review existing literature on video action understanding and text summarization. In Chapter 3, we present our proposed approach for action recognition based on modeling actions as sentences of "action words" and extraction of temporal information from appearances and motions. In Chapter 4, we describe a novel approach for action recognition with a fused spatio-temporal information. In Chapter 5, we present our proposed approach that leverages self-attention networks for skeleton-based action recognition. In Chapter 6, we show our method for an extractive text summarization task based on a mathematical optimization algorithm, DPP and a capsule network which can extract context information from sentences. In Chapter 7, we depict our method for a text summarization task with contextualized representations which benefit from a pre-trained language model. In Chapter 8, we discuss a method of creating sub-sentence highlights for text summarization with DPP. Finally, in Chapter 9, we present our concluding remarks and the lessons learned in this dissertation.

[1] We will release our source code publicly.

# CHAPTER 2
# LITERATURE REVIEW

This chapter reviews the representative works in studying two types of important sequential data, i.e. the literature related to video based action recognition and those related to text summarization. We first present early works on video based action recognition and describe how they obtain temporal cues for the same task. We then describe related works that employ CNNs or recurrent neural networks and two stream networks. We also review similar works trying to overcome the shortcoming of two-stream networks and compare them with our proposed fusion network. In the following section, we depict recent works using the skeleton data for action recognition. We then present latest works that use the transformer network or self-attention network that can extract temporal relation information regardless of their positions. Lastly, we review extractive text summarization methods and mathematical optimization methods including DPP that are exploited for the summarization task. We also discuss recent abstractive text summarization methods based on neural models and compare them with the extractive summarization method.

## 2.1 Temporal Context for Action Recognition and Prediction

Several works using ConvNets to acquire temporal information for action recognition have been studied. In [205], hand crafted features are used in the pooling layer of ConvNet to take advantage of both merits of hand-designed and deep learned features. Temporal information from optical flow is explicitly learned with ConvNets in [164] and the result is fused with the effect of the trained spatial (appearance) ConvNet. [47] merges the ConvNet architecture of the two streams ConvNets [164] to capture spatio-temporal information. Although the aforementioned approaches capture temporal information in small time windows, they fail to capture long-range temporal sequencing information that contain long-range ordered information.

Several works modeling a video-level representation or modeling long temporal information with ConvNets have also been investigated. [48] proposes a method that employs a ranking function to generate a video-wide representation that captures global temporal information. In [182], a HMM model is used to capture the appearance transitions and a max-margin method is employed for temporal information modeling in a video. [36, 182, 133] utilize LSTM [71] units in their ConvNets and attempt to capture long-range temporal information. However, the most natural way of representing a video as long-range ordered temporal information is not fully exploited.

Action prediction is to recognize an action with a partial amount of video data. The task may be considered as a subset of the action recognition problem, in a sense that the input data is limited. [148] proposes the integral BoW and dynamic BoW to model an action in a particular stage. Sparse coding is used to compute activity likelihood of video segments [10]. A max-margin learning

method for prediction is proposed in [10], where human activity is represented in a hierarchical way. [94, 74] employ structured SVM to detect an event and capture global and local dynamics of motions. However, the performance of the above methods are not comparable to our results and they are not applicable to large-scale datasets, such as UCF101 [170].

Our work is inspired by a key idea of sentence classification [235, 79, 81, 90] in NLP. We convert from the domain of images to a domain of words to represent each frame as a word and hence represent a video as a sequence of words, i.e. a sentence. In NLP, words in a sentence are often represented in the form of vectors, see for instance word2vec [126] and Glove [142]. In order to acquire a similar frame-level representation, we adopted the standard BoW [140] encoding method to handle large variability of motions and appearances in video data. It is worth noting, however, that our method can adopt any type of frame-level features to represent video frames as words.

Various ConvNet arichitectures [235, 79, 81, 90] have been taken into account for sentence classification. [81] utilizes dynamic pooling ConvNets for modeling sentences. In [79, 90], a simple 1D ConvNet is employed to classify sentences, and LSTM units are additionally inserted in [235]. Similarly, we utilize a simple but effective ConvNet for learning video word sequencing for action prediction and recognition applicable to large-scale datasets.

## 2.2 Spatio-Temporal Fusion Networks for Action Recognition

Several works using ConvNets to acquire temporal information for action recognition have been studied. In [205], hand-crafted features are used in the pooling layer of ConvNet to take advantage of both merits of hand-designed and deep learned features. Temporal information from optical flow is explicitly learned with ConvNets in [164] and the result is fused with the effect of the trained spatial (appearance) ConvNet. [47] connects several convolution layers of two stream ConvNets to capture spatio-temporal information. Although the aforementioned approaches capture temporal information in small time windows, they fail to capture long-range temporal sequencing information that contains long-range ordered information.

Several works modeling a video-level representation or modeling long temporal information with ConvNets have also been investigated. [48] proposes a method that employs a ranking function to generate a video-wide representation that captures global temporal information. In [182], a HMM model is used to capture the appearance transitions and a max-margin method is employed for temporal information modeling in a video. [36, 182, 133] utilize LSTM [71] unit in their ConvNets and attempt to capture long-range temporal information. However, the most natural way of representing a video as long-range ordered temporal information is not fully exploited.

Recently several researches [88, 106] have used frame level representations for predicting actions with temporal ConvNets. The rational behind these methods is to extract the temporal dynamics more directly by utilizing 1D convolution over time. This approach is widely used in a sentence classification [235, 79, 81] problem in Natural Language Processing literature. Each

word is encoded to vectors and 1D convolution over a sequence of words extracts semantic information between words. For videos, two stream [164] ConvNets are typically employed to train appearance and motion features separately. Once the two streams are trained, sampled RGB or optical flow video frames are fed to each network to extract appearance and motion features respectively. This is the standard feature extraction method and each frame can be represented in a vector form. The biggest advantage of the feature representation is that the temporal information distributed over entire videos can be effectively extracted by using 1D convolutions. Our work is based on the 1D convolution layers to obtain temporal dynamics of appearance and motion cues.

Many ConvNets [166, 201, 237, 191] for image recognition are utilized for action recognition as well. Among them, a concept known as the inception is useful to our encoded data to extract more informative features. The encoded features are convoluted over time with different kernel sizes and concatenated. This process extracts local and global temporal information similar to extracting N-gram semantic information in NLP. [134] introduces an effective residual inception module, which basically has another shortcut connection to the inception module. We employ the residual inception module with 1D convolution layers as it is suitable for extracting temporal dynamics.

The critical drawback of the two-stream [164] ConvNets is the two features cannot be integrated in feature level. In order to solve this problem, different fusion methods are introduced. In [192] they try to extract spatio-temporal features directly by applying 3D convolution to a stack of input frames. [46, 45] connect learned two stream ConvNets to integrate the two stream signals generating the spatio-temporal features. [26] encodes local deep features as a super vector efficiently so

that spatio-temporal information can be handled with spatio-temporal ConvNets. We utilize different basic fusion operations, average, maximum, and multiply, as investigated in [206, 46, 45]. Since we combine the appearance and motion features, we naturally take advantage of two stream ConvNet architecture and connect them with different fusion methods. This work provides a systematic investigation of fusion methods and ablation studies to choose the best fusion methods for better performance.

## 2.3 Skeleton-Based Action Recognition with Self-Attention Networks

Handcrafted features are used to represent the skeleton motion information in early works. [75] computes covariance matrix for joint positions over time. [197] extracts 3D geometric relationships of body parts in Lie group based on rotations and translations of joints. With further progress in deep learning, researchers started using Recurrent Neural Networks (RNN) to extract temporal dynamics between joints as RNNs use sequential processing. [39] proposes a hierarchical RNN that splits the human body into five parts with each part fed into different subnetworks and fuses them hierarchically. [154] splits a cell in an LSTM into part based cells and human body parts are applied to each cell to learn a representation of each part over time. [238] proposes a spatio-temporal LSTM network that learns the co-occurrence features of skeleton joints with a group sparse regularization. [118] introduces trust gate to reduce the influence of noisy joints and employs a spatio-temporal LSTM network to explore the spatila and temporal relationships. [169] introduces attention mechanism in the LSTM network to focus on more important joints at each

time instances. In recent works, CNN based approaches [85, 38, 119, 214] are adopted to learn skeleton features and achieves significant performance. They attempt to convert a skeleton sequence into pseudo images and utilize CNNs to learn. [38] maps a skeleton sequence to a tensor with frames, joints, and xyz coordinates treating it as image and leverages CNNs to train. [85] proposes a method to use relative positions between the joints and the reference joints based on CNNs. [214] maps trajectories of joints to orthogonal planes by using the 2D projection. CNNs are also employed in our method to obtain more informative features from the raw skeleton joints. However, while the aforementioned RNNs and CNNs lack the ability to extract long-term correlation between features, our proposed method fills the gap to obtain high-level semantic information with long-range connections of features.

A self-attention network learns to generate hidden state representations for a sequence of input symbols using a multi-layer architecture [196]. The hidden states of the upper layer are built from the hidden states of the lower layer using a self-attention mechanism. It learns to aggregate information from lower layer hidden states according to their similarities to the $t$-th hidden state. The learned representations are highly effective because they capture deep contextualized information of the input sequence. The self-attentive network with multi-head attention has demonstrated success on a number of tasks including machine translation [196, 181], language modeling and natural language inference [32], semantic role labeling [172], often surpassing recurrent neural networks in terms of accuracy by a substantial margin. Particularly, [196] describes the Transformer model that makes the self-attention mechanism an integral part of the architecture for improved sequence modeling. [32] learns deep contextualized word representations that have led to state-of-the-art

performance on question answering and natural language inference without task-specific architecture modifications. Despite the success, self-attentive networks have not been investigated for the task of human action recognition and in particular skeleton-based action recognition. In this dissertation, we introduce a novel self-attentive architecture to fill this gap.

Temporal information can be extracted from a sequence data such as a video. Many research endeavors have introduced methods for modeling the temporal structure for action recognition [135, 204, 49]. [135] proposes to employ latent variables to decompose complex actions in time and [204] introduces a latent hierarchical model that extends the temporal decomposition of complex actions. [49] utilizes a rank-SVM to model the temporal evolution of Bag of Visual Words (BoVW) representations. [207] introduces a method to model a long-range temporal structure by simply splitting a video into snippets and fusing CNN outputs from each part. We adopt this method since it effectively extracts long-range temporal information and also is applicable to any network with end-to-end training.

## 2.4 Text Summarization with DPP and Contextualized Representations

As a second category of sequential data we study text and in particular text summarization. Extractive summarization approaches are the most popular in real-world applications [13, 29, 52, 72, 227]. These approaches focus on identifying representative sentences from a single document or set of documents to form a summary. The summary sentences can be optionally compressed to remove unimportant constituents such as prepositional phrases to yield a succinct sum-

mary [93, 229, 125, 6, 189, 209, 111, 112, 51, 40]. Extractive summarization methods are mostly unsupervised or lightly-supervised using thousands of training examples. Given its practical importance, we explore an extractive method in this work for multi-document summarization.

It is not uncommon to cast summarization as a discrete optimization problem [58, 178, 115, 70]. In this formulation, a set of binary variables are used to indicate whether their corresponding source sentences are to be included in the summary. The summary sentences are selected to maximize the coverage of important source content, while minimizing the summary redundancy and subject to a length constraint. The optimization can be performed using an off-the-shelf tool such as Gurobi, IBM CPLEX, or via a greedy approximation algorithm. Notable optimization frameworks include integer linear programming [58], determinantal point processes [101], submodular functions [115], and minimum dominating set [157]. In this dissertation, we employ the DPP framework because of its remarkable performance on various summarization problems [230].

Recent years have also seen considerable interest in neural approaches to summarization. In particular, neural extractive approaches focus on learning vector representations of source sentences; then based on these representations they determine if a source sentence is to be included in the summary [19, 226, 127, 130]. Neural abstractive approaches usually include an encoder used to convert the entire source document to a continuous vector, and a decoder for generating an abstract word by word conditioned on the document vector [139, 179, 63, 86]. These neural models, however, require large training data containing hundreds of thousands to millions of examples, which are still unavailable for the multi-document summarization task. To date, most neural summarization studies are performed for single document summarization.

Extracting summary-worthy sentences from the source documents is important even if the ultimate goal is to generate abstracts. Recent abstractive studies recognize the importance of separating "salience estimation" from "text generation" so as to reduce the amount of training data required by encoder-decoder models [56, 108, 107]. An extractive method is often leveraged to identify salient source sentences, then a neural text generator rewrites the selected sentences into an abstract. Our pursuit of the DPP method is especially meaningful in this context. As described in the next section, DPP has an extraordinary ability to distinguish redundant descriptions, thereby avoiding passing redundant content to the abstractor that can cause an encoder-decoder model to fail.

## 2.5 Text Summarization with DPP and Sub-Sentence Highlights

An abstract failing to retain the original meaning poses a substantial risk of harm to applications. Abstractive summarizers can copy words from source documents or generate new words [153, 180, 18, 129, 55, 120, 102]. With greater flexibility comes increased risk. Failing to accurately convey the original meaning can hinder the deployment of summarization techniques in real-world scenarios, as inaccurate and untruthful summaries can lead the readers to false conclusions [12, 44, 97]. In this dissertation, we aim to produce summary highlights which will be overlaid on source documents to allow summaries to be interpreted in context.

Generation of summary highlights is of crucial importance to tasks such as producing informative snippets for search outputs [80], summarizing viewpoints in opinionated text [138, 3], and

Table 2.1: Examples of self-contained and non-self-contained segments extracted from a document sentence.

| **Original Sentence** |
| --- |
| • Some interstates are closed and hundreds of flights have been canceled as winter storms hit during one of the year's busiest travel weeks. |
| **Self-Contained Segments** |
| • Some interstates are closed |
| • hundreds of flights have been canceled as winter storms hit |
| • flights have been canceled as winter storms hit |
| • winter storms hit during one of the year's busiest travel weeks |
| **Non-Self-Contained Segments** |
| • Some interstates are |
| • closed and hundreds of flights have been |
| • been canceled as winter storms hit during one of |
| • hit during one of the year's |

annotating website privacy policies to assist users in answering important questions [150]. Determining the most appropriate textual unit for highlighting, however, has been an understudied problem. Extractive summarization selects whole sentences from documents; a sentence can contain 20 to 30 words on average [82]. Keyphrases containing two to three words are much less informative [65]. Neither are ideal solutions and there is a rising need for other forms of highlighting. We thus investigate sub-sentence highlights that strike a balance between the amount and quality of emphasized content.

It is best for highlighted segments to remain self-contained. In fact, multiple partially-overlapping and self-contained segments can exist in a sentence, as illustrated in Table 2.1. Identifying self-contained segments has not been thoroughly investigated in previous studies. Woodsend and Lapata [220] propose to generate story highlights by selecting and combining phrases; Li et al. [113] explore elementary discourse units generated using an RST parser as selection units; Spala et al. [171] present a crowdsourcing method for workers to highlight sentences and compare systems. Importantly, and distinguishing our work from earlier literature, we make a first attempt to generate self-contained highlights, drawing on the successes of deep contextualized representations and their extraordinary ability of encoding syntactic structure [25, 68].

In the next few chapters, We discuss the methods proposed in this dissertation in greater detail, in the context of two important types of sequential data, i.e. videos of human actions and text in multiple documents.

# CHAPTER 3
# A TEMPORAL SEQUENCE LEARNING FOR ACTION RECOGNITION AND PREDICTION

## 3.1 Approach

In this section, we give a detailed description of the proposed "action word" encoding and "action word" sequence learning. Note that we may often refer to "action words" as simply words in the context of human action recognition. The pipeline of our method is illustrated in Fig. 3.1.

### 3.1.1 BoW Framework for Word Representation

**Feature Extraction**: Since the approaches based on ConvNets [164, 166, 47, 205] recently have achieved competitive results, we utilize *deep-learned* features. In [164], a two-stream ConvNet is trained with stacked optical flows and frames. We follow the two-stream ConvNet method and extract N features $\{\mathrm{x}_1, \cdots, \mathrm{x}_\mathrm{N}\}$, where $\mathrm{x}_t \in \mathbb{R}^D$, every T frame from all videos using the two trained networks. The extracted features are the output vectors of fully connected (FC) layers on both ConvNets and the dimension is $D$. Note that the input frames of consecutive temporal features are overlapped by $(\mathrm{L}-\mathrm{T})$ frames, when $\mathrm{L} > \mathrm{T}$, as we train the temporal network with

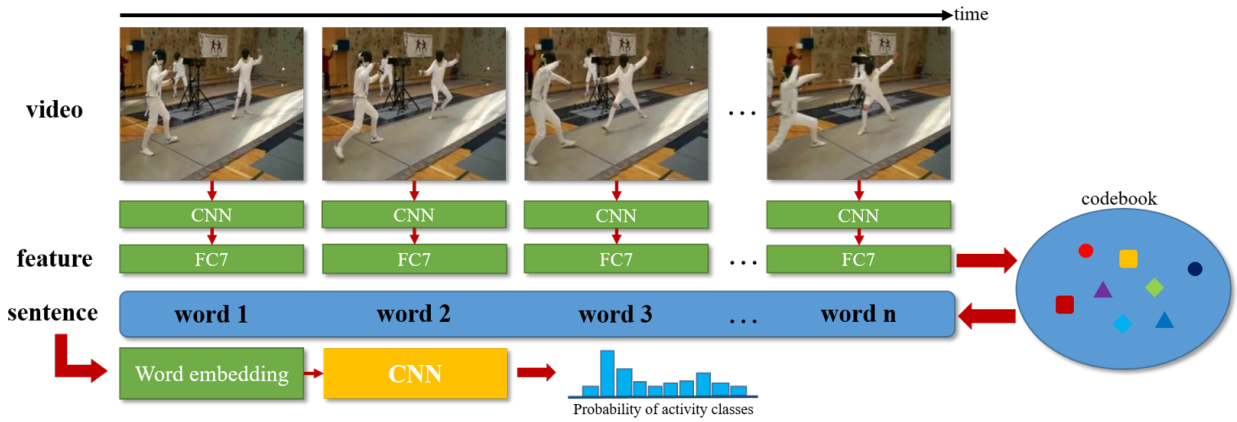



Figure 3.1: Pipeline of our method for action prediction/recognition. First, we extract features from video frames using a trained CNN. We then generate a codebook to assign each feature as *Action Word* as explained in section 3.1.1. Finally, a sequence of *Action Word*s is learned with a sequence learning CNN to classify actions, as described in section 3.1.2.

L stacked frames. The temporal ConvNet is trained with L = 10 and T is set to 5 to consider partial overlap between consecutive temporal features. Also, it should be noted that any frame-wise feature extraction techniques can be utilized to represent each frame as a vector.

**Codebook Generation**: A codebook is generated to represent each feature as an *ActionWord*. A typical choice for constructing the codebook is *k*-means [8] or Gaussian Mixture Model (GMM) [8]. In our method, we used the method of approximate *k*-means [144] to construct the codebook with all extracted features from training videos. The generated $K$ clusters $\{c_1, \cdots, c_K\}$, where $c_k \in \mathbb{R}^D$, are employed to both training and testing videos.

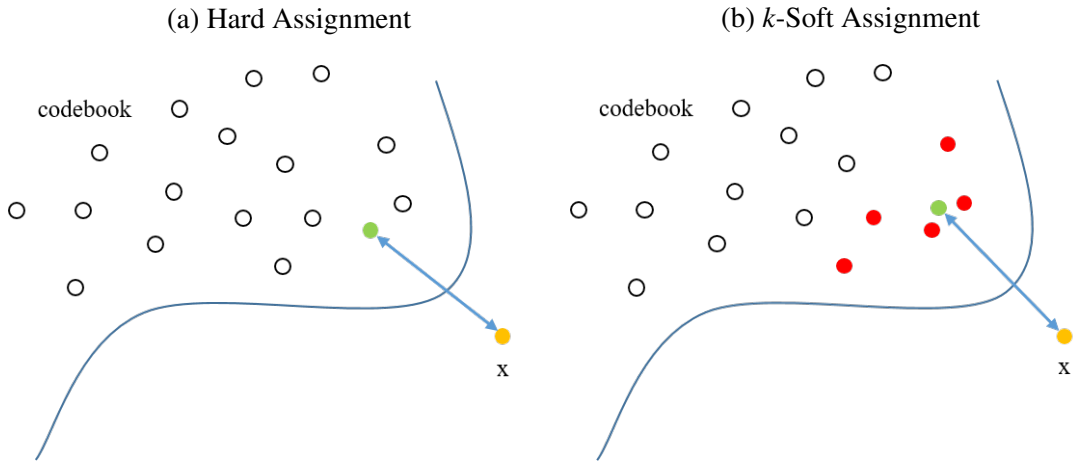



Figure 3.2: Feature encoding methods

**Codeword Assignment**: For coding a video, every extracted video frame feature vector x needs to be mapped to one of the vectors in the codebook, i.e. to one *ActionWord* that best represents the frame-level visual information at time T. We consider two voting based assignment methods: Hard assignment (HA) [167] (or Vector Quantization) and soft assignment (SA) [193], and a direct assignment as described below.

**- Hard Assignment**: With HA, *ActionWord A*, is simply associated with its nearest codeword to the feature as shown in Fig. 3.2a. The nearest codeword is determined as the one best correlated with the feature vector $x_n$. The assigned word number (label) for each feature is a sequential number from 1 to $K$.

$$A_{HA_i} = \arg\min_i \|x - c_i\|_2 \tag{3.1}$$

where $i \in \{1, \cdots, K\}$ and a corresponding weight vector $\omega$ for each feature is associated with one of codewords based on the assigned number.

$$\omega_{\mathrm{x}_{HA}} = \mathrm{c}_i, \quad \text{where } i = A_{HA_i}. \tag{3.2}$$

HA encoding enables reducing memory requirements by maintaining only codewords and the assigned codeword numbers instead of keeping all features. Moreover, the codeword can be ignored and initialized with random values when learning a sequence of assigned numbers. Thus, a video can be represented by a sequence of assigned numbers, leading to memory saving.

**- Soft Assignment**: The SA method considers $k$-nearest codewords to the feature. Fig. 3.2b illustrates an example of 5 nearest neighbor (NN) codewords (5-SA). Five red nearest codewords are correlated with the feature vector x and a weighted centroid vector colored in green is then computed for assignment. The weight vector $\omega$ is computed as follows.

$$\omega_{\mathrm{x}_{SA}} = \sum_{j=1}^{K} \delta(\mathrm{x}, \mathrm{c}_j) \cdot \mathrm{c}_j \cdot d_{\omega_j} \tag{3.3}$$

where $d_{\omega_j}$ is the normalized inverse distance weight:

$$d_{\omega_j} = \frac{\delta(\mathrm{x}, \mathrm{c}_j) \exp(-\beta \left\| \mathrm{x} - \mathrm{c}_j \right\|_2^2)}{\sum_{j=1}^{K} \delta(\mathrm{x}, \mathrm{c}_j) \exp(-\beta \left\| \mathrm{x} - \mathrm{c}_j \right\|_2^2)} \tag{3.4}$$

where $\delta(\mathrm{x}, \mathrm{c}_j)$ is the indicator function for the $k$-NN codewords of x:

$$\delta(\mathrm{x}, \mathrm{c}_j) = \begin{cases} 1, & \text{if } \mathrm{c}_i \in k\text{-NN}(\mathrm{x}), \\ 0, & \text{otherwise.} \end{cases} \tag{3.5}$$

Thus, the computed weight vector $\omega$ gives the weighted centroid of $k$-NN codewords based on inverse distance between the feature and $k$ nearest codewords. Each weight vector $\omega$ is unique, and

therefore an assigned number for each weight vector $\omega$ is also unique. Hence, the total number of assigned numbers is the same as the total number of extracted features in a dataset.

$$A_{SA_i} = i, \quad \text{where } i \in \{1, \cdots, \mathrm{N}\}. \tag{3.6}$$

When learning an *ActionWord* encoded with SA, random vector initialization of the weight vectors cannot be feasible as the assigned numbers are nothing but sequential numbers for each feature. Note that HA can be regarded as a special case of $k$-SA, where $k$ is 1.

**- Direct Assignment**: Instead of computing the codebook, Direct Assignment (DA) encoding considers each video-frame feature as a weighted codeword and assign a unique number to it.

$$\omega_{\mathrm{x}_{DA}} = \mathrm{x} \tag{3.7}$$

$$A_{DA_i} = i, \quad \text{where } i \in \{1, \cdots, \mathrm{N}\}. \tag{3.8}$$

Each frame feature vector is thus directly considered as an *ActionWord*. This method does not require codebook generation leading to reduced computation time, but the memory requirement increases.

### 3.1.2 Sequence Learning with Temporal ConvNet

With the proposed *ActionWord* coding, action prediction and action recognition can be regarded as classification problems for a partial sentence or a sentence. By leveraging the success of sentence classification using ConvNets [235, 79, 81, 90] in NLP, we apply similar ConvNet architectures

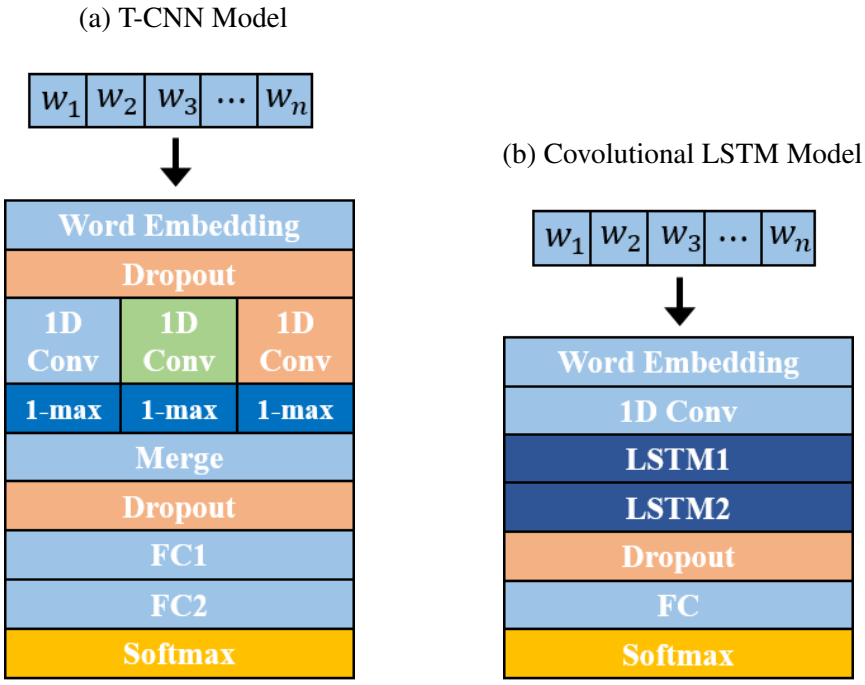



Figure 3.3: ConvNet Architectures

to train and classify *ActionWord* sequences. We consider two ConvNet models: i) T-CNN, ii) Covolutional LSTM (C-LSTM).

**Word Embedding**: The sequence of *ActionWord*s is the input to the ConvNets shown in Fig. 3.3. Since the length of the sequence for each video is different, a word embedding layer is utilized to make the sequences of the same length. The length of each sequence $l_i$ is truncated if $l_i > l_{max}$ whereas $l_i$ is padded with a special codeword that corresponds to $v = [0, \cdots, 0]$ if $l_i < l_{max}$, where $v \in \mathbb{R}^D$ and $l_{max}$ is a user-determined sequence length. The word embedding layer combines the corresponding weight vector $\omega$ based on the assigned word number, and generates an $D \times l_{max}$

matrix for each sequence. The weight vector can be initialized with a random number between -0.05 and 0.05 for the HA random initialization encoding method.

**T-CNN Model**: Fig. 3.3a shows the overall structure of the T-CNN Model. T-CNN consists of $L$ one-dimensional convolution layers denoted by $C^l \in \mathbb{R}^{F_l \times T}$ in parallel where $F_l$ is the number of convolution filters in the $l$-th layer and $T$ is same as $l_{max}$. Each layer consists of temporal convolution, a non-linear activation, and global max (1-max) pooling across time. The collection of filters in each layer is defined as $W = \{W^{(i)}\}_{i=1}^{F_l}$ where $W^{(i)} \in \mathbb{R}^{d \times F_l}$ and a window of $d$ duration. The corresponding bias vector is $b \in \mathbb{R}^{F_l}$. Given the input sequence of weight vectors, $\Omega \in \mathbb{R}^{D \times T}$, the activation $C^l$ is computed such that

$$C^l = ReLU(\mathbb{W} * \Omega + b) \tag{3.9}$$

where $*$ is the convolution operator. The convoluted signals can be viewed as $N$-gram in a sentence, where $N$ can be determined by the size of filters in the convolution layer. After the ReLU activation, the global max pooling is applied to get the largest signal from the activation. Each layer produces a $\nu$ vector where $\nu \in \mathbb{R}^{F_l}$ by concatenating the global max signals. All vectors from $L$ layers are then concatenated generating a $v$ vector where $v = \sum_{i=1}^{L} \nu_i$. The output size of the second FC layer is the number of class in a dataset and Softmax activation is applied in the end.

**Covolutional LSTM Model**: C-LSTM Model consists of a convolution layer and a long short-term memory recurrent neural network (LSTM) [71] designed for time-series data to learn long-term information. Fig. 3.3b shows the overall architecture of the C-LSTM. The multiple parallel convolution layer is not applied because the concatenation of the resulting vectors can break the original sequence for the input of the LSTM layer. The global max pooling layer is also omitted for

Table 3.1: Summary statistics of extracted features for each dataset. $C$: number of classes, $l_{train}$: average sequence length of training data (min / max), $l_{test}$: average sequence length of testing data (min / max), $N$: number of training(testing) sequences(or videos) for each dataset

| | UCF101 | HMDB51 |
|---|---|---|
| $C$ | 101 | 51 |
| $l_{train}$ | 35.8 (4 / 354) | 17.7 (2 / 211) |
| $l_{test}$ | 35.3 (4 / 177) | 17.1 (3 / 128) |
| $N$ | 9537 (3783) | 3570 (1530) |

the same reason. We retain the original order of the sequence and extract more descriptive representations by convolution computation for the sequence. The extracted local temporal information is fed into the LSTM layer and the LSTM layer outputs a video level representation that captures high level temporal information.

## 3.2 Experiments

### 3.2.1 Dataset and Statistics

We test our method on two action video datasets, HMDB51 [99] and UCF101 [170]. The HMDB51 dataset consists of 51 action classes with 6,766 videos and more than 100 videos in each class.

All videos are acquired from movies or Youtube, and contain various human activities, including interactions with other humans or objects. Each action class has 70 videos for training and 30 videos for testing. The UCF101 dataset consists of 101 action categories with 13,320 videos and at least 100 videos are involved in each class. All videos are gathered from Youtube.

Both datasets provide three training and testing splits. We used the first split of each dataset for validating our proposed models. The same parameters and models from split 1 are utilized for other two splits. Table 3.1 shows the statistics of sequence lengths on each dataset for our experiments. We extracted temporal features every 5 frames (T = 5) with 10 stacked input frames (L = 10) and spatial features every 5 frames.

### 3.2.2 Implementation Details

**Training Two-ConvNets**: We use the VGG-16 model [166] for two-stream ConvNets training. Both the temporal and the spatial network are initialized with the pre-trained weights trained with ImageNet [30]. The networks are then fine-tuned with each dataset.

For the training of the spatial network, we use dropout ratios of 0.8 for two FC layers. The input images are resized to make the smaller side as 256. We augment the input images by randomly cropping 224×224 sub-images from the four corners and the center of the original images and randomly flipping in horizontal direction. The learning rate is set to $10^{-3}$ initially and decreased by a factor of 10 when the validation error saturates.

For the training of the temporal network, we use dropout ratios of 0.9 for UCF101 and 0.9 and 0.8 for HMDB51. We pre-compute the optical flows using the TVL1 method [228] before training to improve the training speed. The optical flow input is stacked with L = 10 frames making a 224×224×20 sub-volume. Same data augmentation techniques are employed for the sub-volume and the learning rate is initialized with $5 \times 10^{-4}$ and decreased in the same manner of the spatial network training. A mini-batch of 128 samples are employed at each iteration, but batch normalization method [77] is not used for all trainings.

**Word Vector Representation**: The dimension of temporal $\mathrm{x}_t$ and spatial $\mathrm{x}_s$ feature vectors is 4096. Since the two extracted feature vectors are complementary, we concatenate them with a data ratio $r$, resulting in a combined feature vector x.

$$\mathrm{x} = PCA(\mathrm{x}_{t(1:rD)}) \oplus PCA(\mathrm{x}_{s(1:(1-r)D)}) \tag{3.10}$$

where $D$ is the dimension of x, $0 \leq r \leq 1$, $\oplus$ is a concatenation operation, and $PCA(\mathrm{x}_{1:n})$ is to apply PCA to x and take the first $n$ elements of the projected vector. The reduced dimension of x is $D' \in \{32, 64, 128, 256, 512, 1024\}$. We use the output vector of the penultimate FC (FC7) layer, since the performance with the FC7 vectors is consistently 2∼3% better than the one with the first FC (FC6) layer. In addition, we take the output vector of FC7 with input images or optical flow images that are cropped in the center area making size of 224×224. For the SA and HA feature encoding method, we consider $K = \{5000, 10000, 20000\}$ as the size of codebook.

**Training T-CNN Model for Sequence Learning**: We use three ($L$=3) parallel 1D convolution layers whose filter sizes are 3,4,5 respectively and number of filters are 200. The first dropout rate and the second one are 0.2 and 0.8, respectively. Since the model is simple, we use a somewhat

strong dropout rate to prevent from overfitting. The T-CNN model is trained with a mini-batch size of 64 and the training is terminated after 100 and 300 epochs for UCF101 and HMDB51, respectively.

**Training C-LSTM Model for Sequence Learning**: The filter size of the 1D convolution layer is 5 and its filter count is 200. The number of hidden units of the first and second LSTM layers is 100 and the dropout rate is set to 0.6. Training is terminated after 100 and 200 epochs for UCF101 and HMDB51, respectively. For both models, we use categorical cross entropy loss with Stochastic Grandient Descent and RMSProp [190] step updates, whose learning rate is initialized with $10^{-4}$.

**Tesing**: Given the trained models (T-CNN, C-LSTM), we evaluate the accuracy with the full sequences for the action recognition task, as well as partial sequences for action prediction. Each video sequence is divided into 10 segments creating the following sequences for action predection [148, 94, 103, 74]: 0$\sim$10%, 0$\sim$20%, $\cdots$, 0$\sim$100%.

**Running Time**: The running time of our method is compared with MTSSVM [94], MSSC [10], and Two-stream Fusion [47] methods and the results are listed in Table 3.2. We executed authors' code on a 4.6GHz CPU with 32GB RAM and one TITAN-X GPU. With a sequence of 512-dimension weight vectors, the training time is 51min(T-CNN) and 101min(C-LSTM) on UCF101, and 10min(T-CNN) and 67min(C-LSTM) on HMDB51. Note that the testing time takes a few seconds for each dataset. The T-CNN method is 170$\times$, 507$\times$, 425$\times$ faster than MTSSVM, MSSC, Fusion methods, respectively on UCF101. For the HMDB51 dataset, the T-CNN method is 377$\times$, 1150$\times$, 945$\times$ faster than MTSSVM, MSSC, Fusion methods, respectively. The C-LSTM method

Table 3.2: Training and testing time of comparison methods in hours on UCF101 and HMDB51.

| Methods | UCF101 (hrs) | HMDB51 (hrs) |
|---|---|---|
| MTSSVM [94] | 145 | 83 |
| MSSC [10] | 431 | 253 |
| Fusion [47] (15 epoch) | 362 | 208 |
| Ours (T-CNN) | 0.85 | 0.22 |
| Ours (C-LSTM) | 1.68 | 1.12 |

also spends much less time than compared methods. Note that training time of two-stream ConvNet and feature extraction is not included.

### 3.2.3 Baseline of Two-Stream ConvNets

Table 3.3 shows baseline accuracies for the spatial, temporal, two-stream networks on UCF101 and HMDB51. The value is averaged over three splits and two-stream results are obtain by averaging the prediction probabilities of the spatial and temporal ConvNets. The proposed methods leverage thes baseline two-strema ConvNet and show improvement by taking the temporal information into account.

Table 3.3: Baseline mean performance of spatial, temporal, and two-stream ConvNet on UCF101 and HMDB51. (VGG-16 CNN model is employed.)

| | UCF101 | HMDB51 |
|---|---|---|
| Spatial | 81.8 | 44.8 |
| Temporal | 84.9 | 55.0 |
| Two-stream | 90.1 | 61.4 |

### 3.2.4 Parameter Analysis

**Effects of Dimension and Initialization of Weight Vector**: We first investigate how the weight vector initialization and feature vector size affect the performance. We experiment by setting parameters: with equal data ratios ($r = 0.5$) for temporal and spatial features, with full testing sequences, and with $K = 20$k. Fig. 3.4 shows the results with the T-CNN model. The vectors initialized with weight vectors outperforms randomly initialized weight vectors on both datasets and the performance margin is smaller, as the vector size increases. The randomly initialized vector takes about twice more epochs to be fully trained but data storage can be saved substantially.

In addition, the performance on UCF101 increases as the feature vector dimension increases until 512 with both HA and DA. We speculate this trend occurs because more data is generally helpful but data of size larger than 512 can contain less important data from PCA, so the performance is degraded thereafter. Similar trend happens on the HMDB51 dataset, but no significant performance change is observed between feature vectors of 64 and 512. This means that our

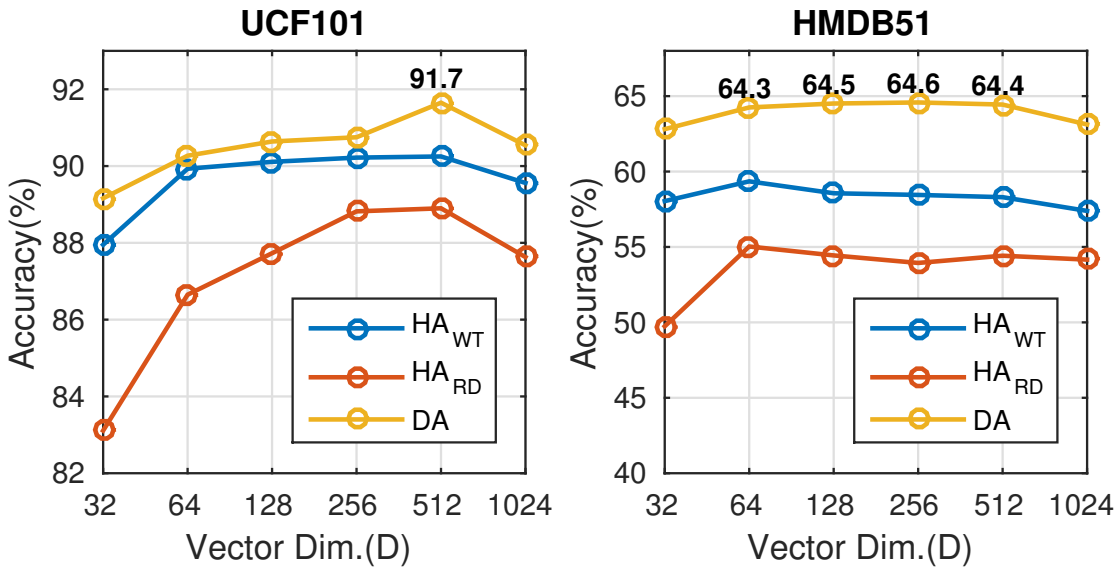



Figure 3.4: Accuracy based on different initialization and dimension of the weight vector $\omega$. $HA_{RD}$ and $HA_{WT}$ denote random initialization and assigned codebook initialization, respectively.

method is robust to the choice of the vector dimension results except the 32-dim vector which loses too much information.

**Effects of Codebook Size and Encoding Methods**. In this experiment, we observe the performance given different codebook sizes and encoding methods. The dimension of the feature vector is fixed to 512, since in the previous experiment the size 512 is found as the most optimal length. The data ratio $r$ is set to 0.5. Fig. 3.5 shows the results with the T-CNN model. The performance of HA decreases as the codebook size increases, while the SA performance increases with larger codebook. In order to investigate these trends, we reduce 128-dimensional 5k and 20k codebooks on UCF101 to 2-dimensional vectors respectively and cluster them with $k$-means, where $k = 101$. We employ the t-SNE dimensionality reduction technique [5], which is well suited for displaying

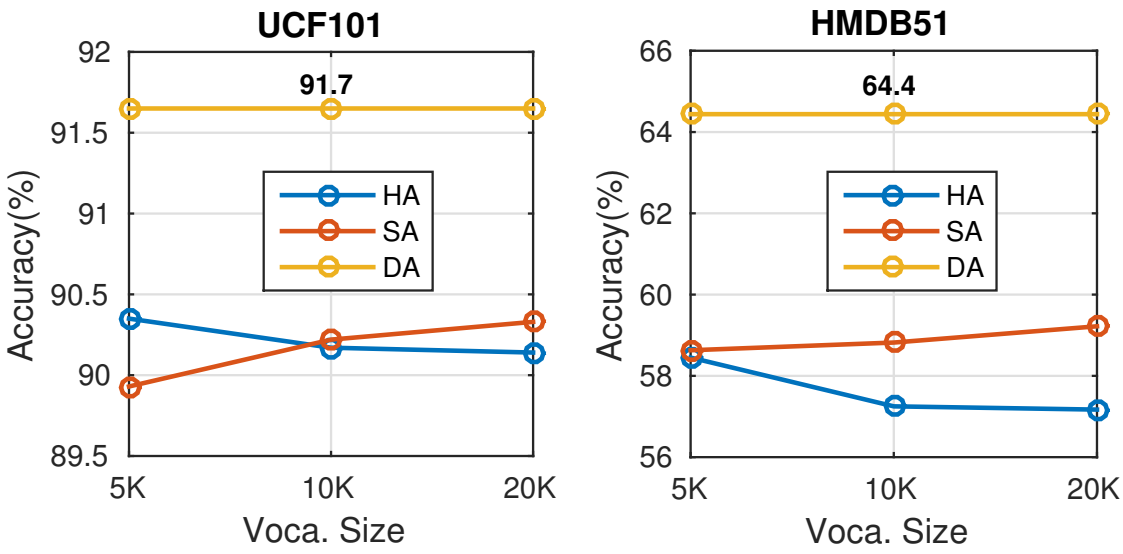



Figure 3.5: Accuracy based on different size of codebook and different encoding methods.

high-dimensional data. As shown in Fig. 3.6, the 5k codebook has larger margin between clusters than the 20k codebook. Therefore, with HA, it is less likely to mislabel with the 5k codebook than the 20k codebook. On the other hand, with SA, the 5 NN codebooks can group more tightly with the 20k codebook, so the centroid of 5NN is likely to be closer to the original feature vector than the centroid in the 5k codebook. In any cases, since the performance gain of different codebook sizes is small, we can argue that our method is robust to the choice of the codebook size. Another distinctive observation is that DA outperforms other encoding methods with relatively large margin.

(a) 5k Codebook (b) 20k Codebook

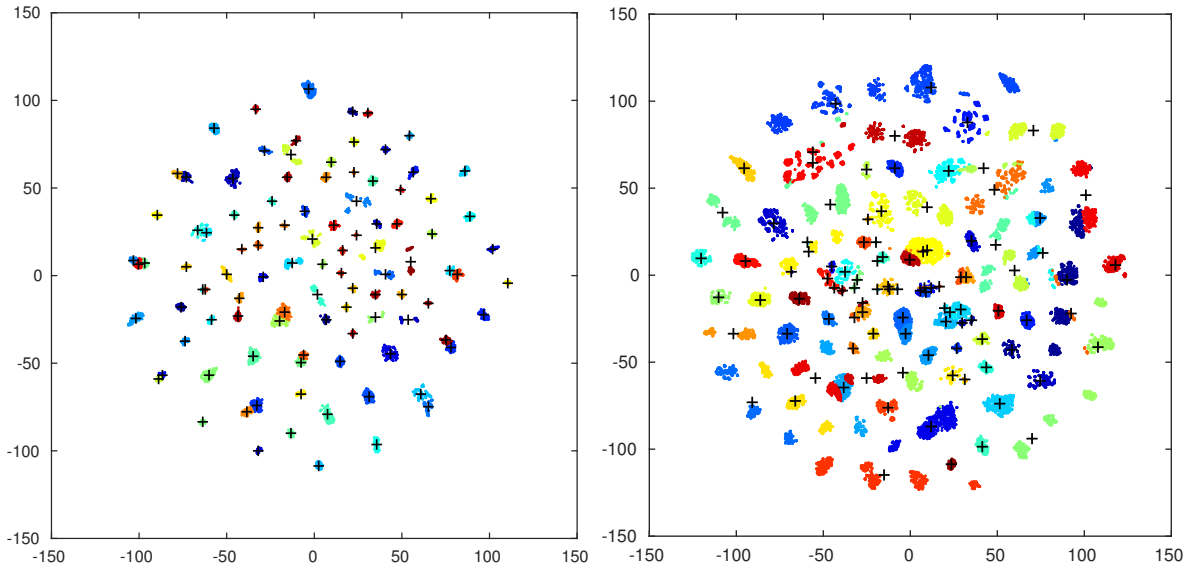



Figure 3.6: Visualization of 5k and 20k codebooks ($D = 2$) of UCF101. Each codebook is clustered with $k$-means ($k = 101$).

### 3.2.5 Optimal Data Ratio

The temporal and spatial feature vectors are concatenated based on the data ratio $r$ in eq. (3.10). As shown in Table 3.3, the temporal network outperforms the spatial network on both datasets. In this analysis, we empirically find an optimal ratio that assigns higher weight to the temporal feature vector.

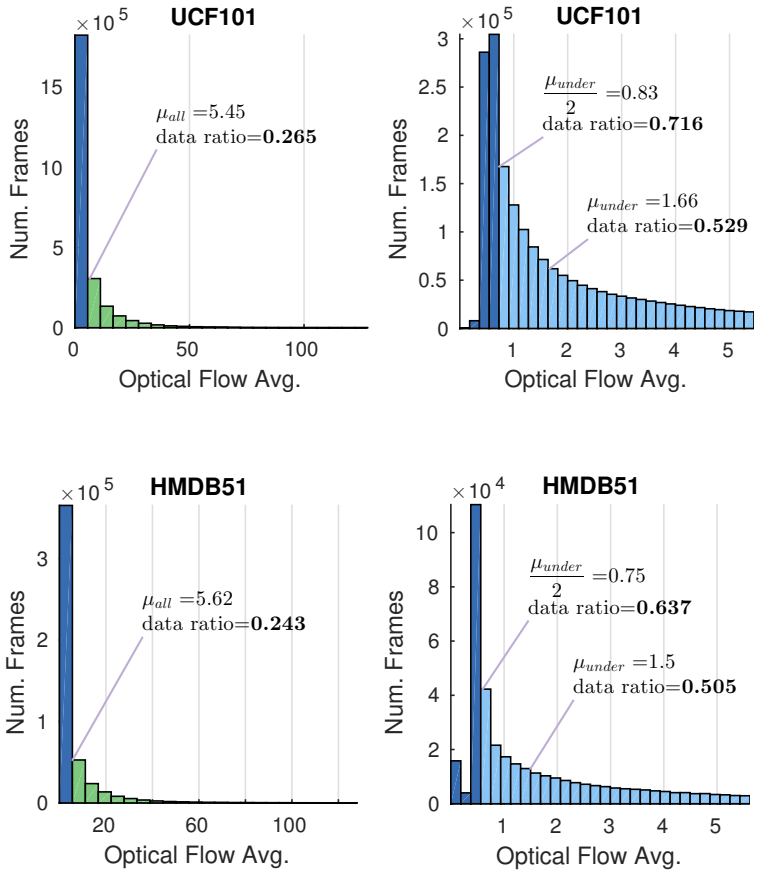



Figure 3.7: Histogram of average optical flow on UCF101 and HMDB51.

First, we compute the frame-wise average of optical flow magnitudes along the two axis as follows:

$$f_i = \frac{1}{2}\left(\Sigma_{k=1}^{P} abs(f_{u_{i,k}} - 128)/P + \Sigma_{k=1}^{P} abs(f_{v_{i,k}} - 128)/P\right)$$

where $f_i$ is the average optical flow for the i-th frame in the video, $P$ is the total number of pixels in the i-th frame, and $f_u$, $f_v$ are the horizontal and the vertical optical flow values, respectively. Of course, the intuition is that frames with higher motion information can be identified using $f_i$.

Table 3.4: Performance based on different data ratios and feature dimensions on HMDB51 and UCF101 split 1.

| HMDB51 | $r = 0.5$ | $r = 0.625$ | $r = 0.75$ |
|---|---|---|---|
| 64 | 65.2 | 65.2 | 66.0 |
| 128 | 65.0 | 65.6 | 65.0 |
| 256 | 64.6 | 65.7 | 64.6 |
| 512 | 64.8 | **66.4** | 65.1 |
| UCF101 | $r = 0.5$ | $r = 0.625$ | $r = 0.75$ |
| 512 | 91.5 | 91.8 | **92.7** |

We explain the choice of $r$ using the histograms of $f_i$ shown in Fig. 3.7. The left column shows that the frames in the green colored bins contain more motion cues than the frames in the blue colored bins. Also, majority of the frames fall below the mean of the $f_i$ across all frames, i.e. $\mu_{all}$. These are frames that contain less motion information, and hence provide more spatial appearance information. A first order estimate of $r$ could then be given by the ratio of frames above $\mu_{all}$ over total number of frames. However, since motion is a stronger cue,it is reasonable to assume that better estimates of $r$ would be given by the first quartile or the half of the first quartile. Therefore, consider the graphs on the right column of Fig. 3.7, which show the histograms of $f_i$ only for frames whose average optical flow is smaller than $\mu_{all}$. We compute the mean of these lower histograms, denoted as $\mu_{under}$, which determine the first quartile of the original histogram.

Table 3.5: Action recognition performance comparison with State-of-the-art. (mean over three splits)

| HMDB51 | | UCF101 | |
|---|---|---|---|
| iDT+FV [201] | 57.2 | iDT+FV [136] | 85.9 |
| iDT+HSV [140] | 61.1 | iDT+HSV [140] | 87.9 |
| VideoDarwin [48] | 63.7 | LRCN [36] | 82.9 |
| Two stream [164] | 59.4 | Two stream [164] | 88.0 |
| TDD+FV [205] | 63.2 | TDD+FV [205] | 90.3 |
| KVMF [237] | 63.3 | KVMF [237] | **93.1** |
| Fusion [47] | 65.4 | Fusion [47] | **92.5** |
| Transformation [215] | 62.0 | Transformation [215] | 92.4 |
| Ours(C-LSTM) | 62.4 | Ours(C-LSTM) | 90.9 |
| Ours(T-CNN) | **66.3** | Ours(CNN) | **92.5** |

Better estimates of the ratio $r$ are then given by the ratio of frames above $\mu_{under}$ or $\mu_{under}/2$ over the total number of frames.

In our experiments, we found that the ratio $r$ given by $\mu_{under}$ is 0.529 on UCF101 and 0.505 on HMDB51 meaning that $\mu_{under}$ is close to median of the average optical flows. The estimate based on $\mu_{under}/2$, resulted in $\sim$0.75 for UCF101 and $\sim$0.625 for HMDB51. One observation is that the UCF101 dataset involves many sports and exercise videos [66] that generally contain larger

Table 3.6: Action Prediction performance on UCF101 and HMDB51.

| UCF101 | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|
| MOS [74] | – | 35.0 | – | 37.1 | – | 39.4 | – | 40.3 | – | 40.9 |
| SMMED[74] | – | 40.6 | – | 40.6 | – | 40.6 | – | 40.6 | – | 40.6 |
| Fusion [47] | **82.8** | 85.5 | 87.5 | 88.8 | 89.2 | 90.4 | 90.7 | 91.0 | 91.5 | 92.5 |
| Ours | 82.2 | **86.7** | **88.5** | **89.5** | **90.1** | **91.0** | **91.5** | **91.9** | **92.4** | **92.5** |
| HMDB51 | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
| Fusion [47] | **44.8** | 51.5 | 54.5 | 58.0 | 61.0 | 62.9 | 64.9 | 65.2 | 65.4 | 65.4 |
| Ours | 38.8 | **51.6** | **57.6** | **60.5** | **62.9** | **64.6** | **65.6** | **66.2** | **66.3** | **66.3** |

motions, while the HMDB51 dataset consists of simple action videos [66] that have moderate motion. The ratios computed with $\mu_{under}/2$ support this observation. Results using DA and T-CNN for these ratios are shown in Table 3.4. The best performance is achieved with estimates based on $\mu_{under}/2$, confirming that the estimated ratios are reliable.

### 3.2.6 Action Recognition Performance

Table 3.5 shows action recognition results of recent state-of-the-art methods. Our best result outperforms other methods by 0.9% on HMDB51 and is compatible on UCF101. We conjecture that [237] outperforms ours because they utilize GoogLeNet [177] with batch normalization [77], which is a deeper network than VGG-16 [166]. Our result is on par with Fusion [47] on UCF101

but its computational efficiency is much better due to the fast-trainable network as shown in Table 3.2. The C-LSTM model, however, does not learn much comparing with the baseline accuracy. We speculate this is because the temporal 1D convolution without pooling does not represent a video effectively. Applying 1D convolution followed by max pooling over several small segments may boost the performance for the C-LSTM model.

### 3.2.7 Action Prediction Performance

The goal in action prediction is the same as in action recognition, except that the input test video is not a full video. Our method can take a variable size input so the partial input can be readily handled. In order to compare with a method using T-CNN, we evaluate Fusion [47] with the partial test video frames. We follow their testing procedure by taking 5 uniformly spaced frames from the given range. The horizontally flipped input frame is augmented and the entire frame is used.

Table 3.6 show the action prediction results with comparing methods. Our results consistently outperform the Fusion method as well as the previous best results: MOS and SMMED [48]. We observe an interesting trend, in the sense that our result is only outperformed by Fusion in the first 10% range. We conjecture two reasons about the result: the length of the sequence is too short to be fully trained, and noisy words are inserted to the sequence especially on HMDB51. On the other hand, our method rapidly reaches to full accuracy with partial data. The prediction results with half-video data reach 95% and 97% of full accuracy for the HMDB51 and UCF101, respectively.

Also, the performance with 90% of frames is almost identical to full accuracy. These observations show that our method is well suitable to detect actions with partial data.

## 3.3 Conclusion

We proposed an effective and efficient sequence learning method that captures global temporal sequencing information of a video. This is achieved by means of a new video representation as a sequence of visual words (a sentence). By training a ConvNet to learn the sequences corresponding to different actions, we are able to accurately identify an action or predict it from a partial sentence. The ConvNet architecture is simple and can be trained with minimum computational cost. We also demonstrate how important hyper-parameters such as data ratio are determined automatically. These parameters play significant roles in improving the accuracy. We achieve compatible state-of-the-art results on both action recognition and action prediction.

# CHAPTER 4
# SPATIO-TEMPORAL FUSION NETWORKS FOR ACTION RECOGNITION

## 4.1 Approach

A video contains many redundant temporal information between consecutive frames. Instead of densely sampled feature points [202], [208] samples frames in different video segments, while [164] deals with multiple consecutive frames. These techniques train ConvNets for different modalities, appearance and motion, and use late fusion to combine them. However, two issues are raised from these methods: (1) multiple consecutive frames only cover local temporal dynamics not global temporal dynamics over videos, and (2) the prediction score fusion only captures dynamic of each appearance and motion cue separately not the spatio-temporal dynamics. In this section, we propose a spatio-temporal fusion network (STFN) to extract temporal dynamic information over an entire video and combine appearance and motion dynamics, using end-to-end ConvNets training, as shown in Fig. 4.1. The network has the following properties: (1) convolutions are computed over time so that the temporal dynamic information is extracted; (2) each convolution block extracts local and global temporal information with different feature map sizes; and (3) the extracted appearance and motion dynamic features are integrated through an injection

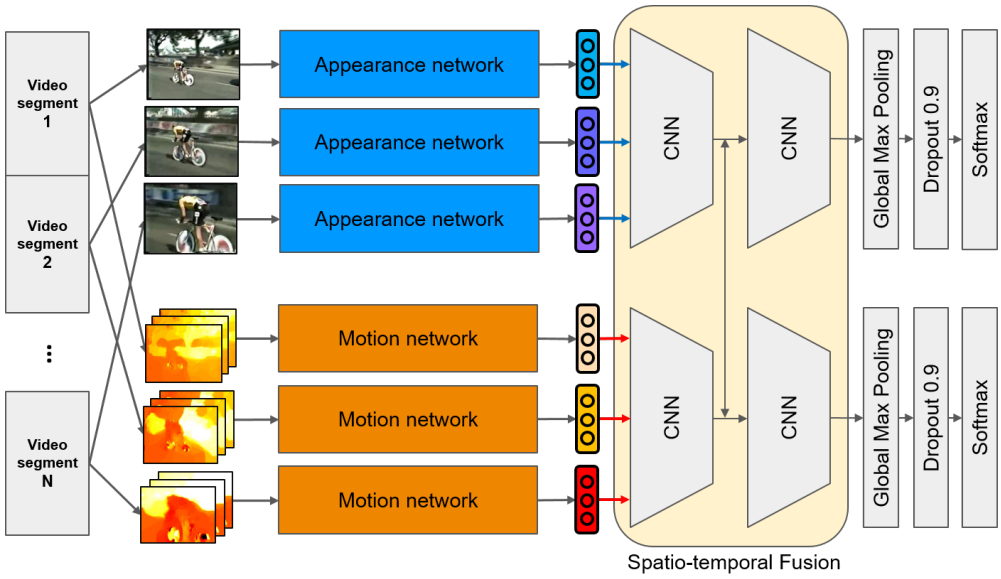



Figure 4.1: The proposed spatio-temporal fusion network. The number of segments is an arbitrary number. We use three segments in the figure for illustration purpose.

from one to the other or with bi-direction way. More details about STFN are described in Section 4.1.1.

### 4.1.1 Spatio-Temporal Fusion Networks

We consider the output feature maps of CNNs for $N$ segments from a video $V$. Each feature map $\{F_1, F_2, \cdots, F_N\}$ is a vector of size $F \in \mathbb{R}^d$, where $d$ is the output feature map dimension. The feature maps can be retrieved from different networks trained with different modalities such as appearance and motion. $F^a, F^m$, where $F^{\mathrm{x}} \in \mathbb{R}^{N \times d}$, are the feature maps from appearance and

motion networks, respectively. STFN is applied to the sequence of feature maps, $F^a$ and $F^m$, to extract temporal dynamics of each feature map and fuse them as follows:

$$\begin{aligned}\text{STFN}(F^a, F^m) = \mathscr{H}(\mathscr{F}(\mathscr{G}(\mathscr{F}(F^a; \text{W}_\text{a}), \mathscr{F}(F^m; \text{W}_\text{m})); \text{W}_\text{fa})) + \\ \mathscr{H}(\mathscr{F}(\mathscr{G}(\mathscr{F}(F^a; \text{W}_\text{a}), \mathscr{F}(F^m; \text{W}_\text{m})); \text{W}_\text{fm}))\end{aligned} \tag{4.1}$$

$\mathscr{F}(F^\text{x}; \text{W}_\text{x})$, where $\text{x} \in \{a, m, fa, fm\}$ meaning appearance, motion, fused appearance, fused motion sequences, is a ConvNet function with parameters $\text{W}_\text{x}$ which produces sequences of same input sizes for the given sequences. More details about the ConvNet are given in Section 4.1.2.1. The fusion aggregation function $\mathscr{G}$ combines the output sequences of appearance and motion dynamic information. $\mathscr{G}$ and the follow-up ConvNets, $\mathscr{F}(F^\text{x}; \text{W}_\text{fa})$, can be omitted depending on the design choice of STFN. More details are provided in the next subsection. From the learned sequences, the prediction function $\mathscr{H}$ predicts the probability of each activity class. Softmax function, which is widely used for multi-class classification, is chosen for $\mathscr{H}$.

The overall network is learned in an end-to-end scheme like TSN [208]. The sequences of feature maps are $X = F^a, F^m$ and the outputs of the $\mathscr{F}$ function are denoted by $y$. Also, let $\mathscr{L}$ be the loss function. The gradient of the loss function with respect to $X$, $\frac{d\mathscr{L}}{dX}$, during the training process is defined as:

$$\frac{d\mathscr{L}}{dF_k^\text{x}} = \mathscr{F}(F_{k'}^\text{x}; \text{W}_\text{x}) \frac{d\mathscr{L}}{dX} \tag{4.2}$$

where $k \in N$ and $k' = \{1, 2, \cdots, k-1, k+1, \cdots, N\}$. In the end-to-end training, the parameters for the $N$ segments are learned using stochastic gradient descent (SGD). The parameters are learned from the entire video with segmented temporal inputs.

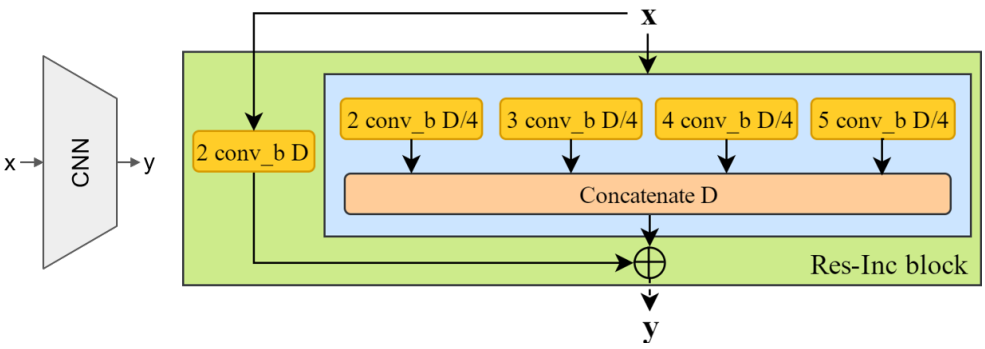



Figure 4.2: A Residual Inception block. The res-inc block in the right figure shows the components of the CNN in the left figure. The number in each module inside of the Res-Inc block depicts convolution kernel size. conv_b consists of the 1D convolution, batch normalization, and relu activation layers. $D$ represents the input vector dimension, $d$.

### 4.1.2 STFN Components

In this subsection, we describe the ConvNets, $\mathscr{F}$, and the fusion aggregation function, $\mathscr{G}$, in detail. We also discuss different STFN architectures to find the most suitable model.

#### 4.1.2.1 Residual Inception Block

A sequence of frame representations, $F^a, F^m$, inherently contains temporal dynamics between features. The consecutive features are convoluted over time with different kernel sizes to extract local temporal information. This operation is conceptually similar to an n-gram of a sentence that contains local semantic information among n words. The convoluted features are then concatenated to

formulate a hierarchical feature from each input. Motivated by an inception module [191, 134] that convolves an input signal with different filters, we design an inception block with different kernel sizes as shown in Fig. 4.2. The input signal $F^{\mathrm{x}}$ is convoluted across time using 1D convolution with four different sizes of kernels, 2,3,4,5, whose filter size is a quarter of the input dimension, $d$. The 1D convolution retains the same temporal length as the input. We did preliminary experiments to find out the best combination of the kernel sizes and 2,3,4,5 shows the best performance. We designed the filter size of each convolution to be a quarter of input dimension, making the concatenated feature have the same dimension as the input with same weight. We also used convolution layers with kernel size of 1 [191, 134] before the conv_b block to reduce the input dimension. However, they decrease the performance since it perturbs the input signal that contains temporal dynamics, so we decided not to include them.

The concatenated multi features and the input signal $F$ are added for residual learning [36]. We chose a convolution kernel size of 2 for the skip connection to capture the smallest local temporal information. Formally, the Residual Inception (Res-Inc) block in this paper is defined as:

$$\mathrm{y} = \mathscr{C}(\mathscr{R}(F^{\mathrm{x}}, \{W_i\})) + \mathscr{R}(F^{\mathrm{x}}, \{W_j\}) \tag{4.3}$$

where $\mathscr{R}$ is the convolution function with weights $W_i, i \in \{2,3,4,5\}$ for the residual connection or $W_j, j \in \{2\}$ for the skip connection, and the function $\mathscr{C}(\cdot)$ represents a concatenation operation. In Fig. 4.2, x is identical to $F^{\mathrm{x}}$ in Equation 4.3. The convolution block, conv_b, is composed of Batch normalization [136] and ReLU [191], while the convolution block in skip connection lacks the ReLU activation layer. The output signal is further activated with ReLU before it is aggregated with the other signal. The output sequence of the Res-Inc block contains more discriminative

temporal dynamic information than the input sequence. Since the Res-Inc block outputs signals of same dimension of input signals, a series of Res-Inc block can be easily setup.

#### 4.1.2.2 Spatio-Temporal Fusion

Despite the successful performance with the two-stream approach, a clear drawback is that a spatio-temporal information is not achievable with separate training of the appearance and motion data. The appearance and motion information are complementary to each other in order to discern an action of similar motion or appearance patterns e.g. brushing teeth and hammering. In order to overcome this deficiency, a number of researches have been looking into fusing two-stream networks [46, 45, 47] directly and learning spatio-temporal features [192, 188] . Although, their results show improved performance, their spatio-temporal features are limited to local snippets of an entire video sequence. In contrast, STFN takes advantage of extracted temporal dynamic features that capture long term temporal information over entire video to fuse them.

We investigate three different fusion operations $\mathscr{G}$ with the output sequences of two Res-Inc blocks $\{P_1^{\mathrm{x}}, P_2^{\mathrm{x}}, \cdots, P_N^{\mathrm{x}}\}$, where $P_n^{\mathrm{x}} \in \mathbb{R}^d$ , and $\mathrm{x} \in \{a, m\}$ represent either appearance or motion features.

**Element-wise Average**

$$P_n' = \frac{(P_n^a + P_n^m)}{2} \tag{4.4}$$

where $P'$ is the aggregated sequence and $n \in \{1, 2, \cdots, N\}$. This operation leverages all information and uses the mean activation for the fused signal. This operation may get affected by noisy input signals but since we deal with highly informative features, it is a good choice for our architecture.

**Element-wise Multiplication**

$$P'_n = P^a_n \times P^m_n \tag{4.5}$$

The intuition behind this operation is to amplify a signal when both signals are strong, i.e. similar to attention mechanism. However, the noisy strong signal may affect heavily the fused signal leading to performance decrease.

**Element-wise Maximum**

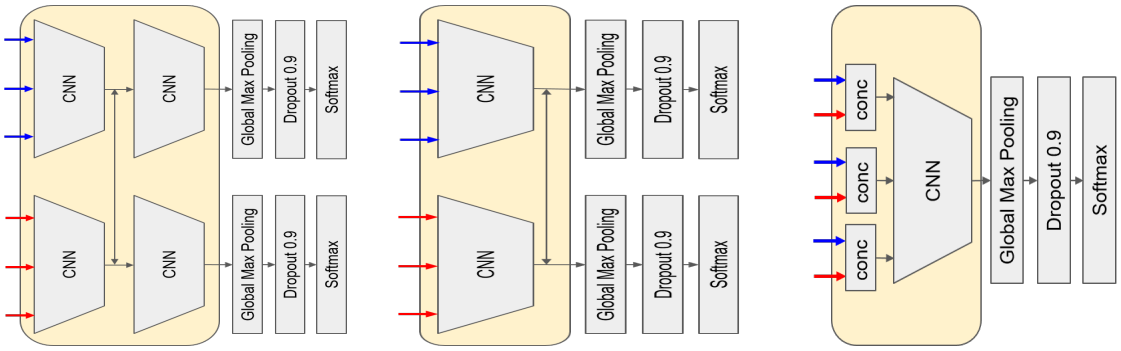
$$P'_n = max(P^a_n, P^m_n) \tag{4.6}$$

The idea of max pooling is to seek the most discriminative signal among inputs. It selects either appearance or motion cue for each element of input signals. This operation may confuse the following Res-Inc block since the aggregated vectors are mixed with the appearance and motion signals.

We compare the performance of each operation in the ablation studies.

#### 4.1.2.3 Architecture Variations of STFN

We propose different design architectures of STFN and investigate them in detail. Fig. 4.3b is a variation of Fig. 4.3a where we want to learn how the additional Res-Inc blocks affect to the results.




(a) (b) (c)

Figure 4.3: Different designs of spatio-temporal fusion architecture. (a) shows our proposed architecture; (b) lacks the follow-up Res-Inc blocks after fusion; and (c) concatenation of the appearance and motion sequences in feature level before extracting temporal dynamics. The blue and red arrows represent the appearance and motion sequence inputs, respectively.

The Res-Inc blocks after fusion extract temporal dynamics of spatio-temporal features leading to better performance. In Fig. 4.3c, fusion is executed in feature-level by simply concatenating appearance and motion signals. This fused signal is fed to the Res-Inc block to extract temporal dynamic information.

#### 4.1.2.4 Fusion Direction

As shown in Fig. 4.4, aggregating two signals can be three possible ways: appearance to motion, motion to appearance, and bi-directional fusion. The fused signals are fed to the next Res-Inc blocks and affect to the residual and skip connection along the forward an backward propagations

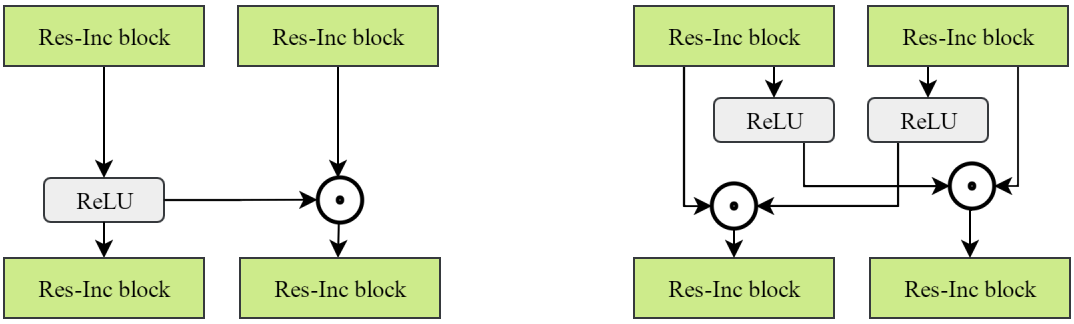



Figure 4.4: Two types of fusion methods: asymmetric and symmetric fusion. (a) shows asymmetric fusion method and two fusions are possible with this method: appearance to motion features and motion to appearance features. (b) shows symmetric fusion where each fused signal is further used in following layers. Two signals are merged with the previous described fusion operations. Note that this figure only illustrates the fusion connections between two Res-Inc blocks and the rest layers are omitted.

when training. Considering the three fusion operations, only multiplication operation results in byproduct signal from partial derivatives of the fused signals when signals are back-propagated. This means the fusion with multiplication operation makes the input signal change rapidly than other operations. Thus, it is not easy to learn proper spatio-temporal features especially when there is significant gap between the discriminative abilities of appearance and motion features.

## 4.2 Experiments

In this section, we first discuss the datasets and implementation details. Then we evaluate each design choice for STFN. Finally, we compare our best performance with the state-of-the-art methods.

### 4.2.1 Datasets

We tested our method on two large action datasets, HMDB51 [99], UCF101 [170]. The HMDB51 datset consists of 51 action classes with 6,766 videos and more than 100 videos in each class. All videos are acquired from movies or youtube and contain various human activities and interactions with human or object. Each action class has 70 videos for training and 30 videos for testing. The UCF101 dataset consists of 101 action categories with 13,320 videos and at least 100 videos are involved in each classes. UCF101 provides large diverse videos with a fixed resolution of $320 \times 240$ with 5 different types of actions. All videos are gathered from youtube. Both datasets provide evaluation scheme for three training and testing splits and we follow the original evaluation method.

### 4.2.2 Implementation Details

**Two-stream ConvNets**: ResNet-101 [36] and Inception-V3 [83] are employed for the base networks to train appearance and motion networks. Both networks are initialized with the pre-trained weights trained on the ImageNet [30] dataset. To fine-tune the networks, we replace the classification layer with $C$ softmax layer, where $C$ is the number of action classes. The appearance network takes RGB images, while the motion network a stack of 10 dense optical flow frames. The input RGB or optical flow images are resized to make the smaller side as 256. We augment the input image by cropping, resizing, and mirroring in horizontal direction. The width and height of the cropped image are randomly sampled from $\{256, 224, 192, 168\}$, and the input images are cropped from the four corners and the center of the original images. The cropped images are then resized to $224 \times 224$ for the network input. This augmentation considers both scale and aspect ratio. We pre-compute the optical flows using the TVL1 method [228] before training to improve the training speed. The optical flow input is stacked with 10 frames making a $224 \times 224 \times 20$ sub-volume for $x$ and $y$ directions. Same data augmentation techniques are employed for the optical flow sub-volume. We use mini-batch stochastic gradient descent (SGD) to learn models with a batch size of 32 and momentum of 0.9. The learning rate is set to $10^{-3}$ initially and decreased by a factor of 10 when the validation error saturates, for both networks.

**STFN**: In order to train STFN, we retain only convolutional layers and global pooling layer of each network, similar to [34]. The feature maps for STFN are extracted from the output of the global pooling layer. The output dimension is 2048 for both ResNet-101 and Inception-V3.

We apply two step training process. We first fix the weights of trained appearance and motion networks and train only STFN. Then we train the entire networks with same methods described in Two-stream ConvNets training. For the first training, we initialize the learning rate with $10^{-4}$ and decrease it until $10^{-7}$ by a factor of 10 when the validation error saturates. RMSProp [190] optimizer is used for the STFN training. The second training is executed with same setting of the two-stream ConvNets training without fixing all weights. For training and testing, we divide the videos into $N = 5$ segments with same lenghts. Note that we use $N = 5$ for all evaluation except for the experiment in Section 4.2.6. A random frame is selected from each $N$ segment and optical flow stacks centered on the selected frames are associated for two input sequences. We apply same augmentations for selected frames and optical flow stacks in an input sequence. When testing, 5 frames are uniformly sampled from each segment making 5 sequences and the final prediction scores are averaged over each output. The experiments are performed with 5 segments, average fusion operation, and bi-directional fusion as defualt except for each ablation study.

### 4.2.3 Evaluation of Different Designs

As we discussed in Section 4.1.2.3, the performances of three proposed STFN architectures are presented in Table 4.1. Comparing Fig. 4.3a and Fig. 4.3b networks, we verify that the Res-Inc blocks make important role extracting temporal dynamics. We conjecture that the consecutive Res-Inc blocks extract temporal dynamics of fused features and they contain better video-wide discriminative features. Another architecture design, Fig. 4.3c, is introduced to see how the feature

Table 4.1: Prediction accuracy(%) on the first split of HMDB51 and UCF101 using different architectures of STFN as shown in Fig. 4.3.

| Design | HMDB51 | UCF101 |
|---|---|---|
| Fig. 4.3a | **70.4** | **93.5** |
| Fig. 4.3b | 69.6 | 93.2 |
| Fig. 4.3c | 69.2 | 92.0 |

level fusion affects to the performance as opposed to the baseline two-stream networks. We observe the significant performance drop in both datasets and it proves the importance of the fusion scheme. Since the architecture of Fig. 4.3a shows the best performance, we choose it as our default STFN network.

The result with a single Res-Inc module (4.3b) outperforms the baseline late fusion results shown in Table 4.5 by 8.1% on HMDB51 and 0.2% on UCF101. This shows the effectiveness of the Res-Inc module. With another Res-Inc module and feature fusion, 0.8% and 0.3% additional gains are obtained on HMDB51 and UCF101, respectively. Note that from a preliminary experiment by increasing the number of consecutive Res-Inc blocks from two to four, we observe performance drops: 3.8%, 6.5% on HMDB51, 4.1%, 6.9% on UCF101. The signals undergo the Res-Inc block contain temporally convoluted information with different kernel sizes (from residual connection). More Res-Inc blocks extract higher level temporal information, but we conjecture that signals experienced more than two levels confuse the original temporal orders, introducing noise.

Table 4.2: Prediction accuracy(%) on the first split of HMDB51 and UCF101 using different fusion operations.

| Fusion operation | HMDB51 | UCF101 |
|---|---|---|
| Average | **70.4** | **93.5** |
| Maximum | 69.5 | 92.9 |
| Multiplication | 68.3 | 92.6 |

### 4.2.4 Evaluation of Fusion Operations

This section presents the performances based on different fusion operations: Element-wise average, maximum, and multiplication. As shown in Table 4.2, the average operation outperforms other methods. It is interesting to see the performance gap between the average and the multiplication operations, 0.9% and 2.1% for HMDB51 and UCF101 respectively. We speculate the reason is due to the performance discrepancy of two networks as shown in Table 4.5. With multiplication, the inferior feature (appearance cue on HMDB51) could harm the fused signal. Also, it is better to take into account all data by averaging than picking the strongest signals since STFN deals with highly pre-processed signals. From the results, we take the average operation as our default choice. Note that we tried weighted average based on the normalized performances of baseline networks and automatic scaling by appliying 1x1 2D conv to each signal before fusing. However simple average results in the best performance.

Table 4.3: Prediction accuracy(%) on the first split of HMDB51 and UCF101 using different fusion directions. A and M represent the appearance and motion features, respectively. The bottom two methods are asymmetric fusion methods whereas the top one is bi-direction fusion method.

| Fusion direction | HMDB51 | UCF101 |
|---|---|---|
| A↔M | **70.4** | **93.5** |
| A←M | 70.3 | 93.4 |
| A→M | 70.1 | 93.2 |

### 4.2.5 Evaluation of Fusion Directions

In Table 4.3, we compare the performance variation with different fusion directions. Note that A←M is a simply reflected network of A→M and we use 5 segments for all experiments. For the asymmetric fusion methods, A←M connection outperforms the other way consistently on both datsets. This effect is due to the fact that the motion stream overfits quickly with the A→M fusion and no further spatio-temporal learning occurs. This comes from the base performance different between appearance and motion features so that fusion injection to the higher discriminative feature leads to worse performance. The bi-direction fusion outperforms A←M with small margin, 0.1% on HMDB51 and 0.2% on UCF101. This makes sense since two spatio-temporal features are learned simultaneously in two streams, whereas asymmetric fusion learns spatio-temporal in the injected stream and the learned weights are propagated to the other stream only when back propagating from the fused connection. However, we argue that our proposed STFN is robust to

Table 4.4: Prediction accuracy(%) on the first split of HMDB51 and UCF101 using different numbers of segments in videos.

| Number of segments | HMDB51 | UCF101 |
|---|---|---|
| 3 | 70.3 | 93.2 |
| 5 | 70.4 | 93.5 |
| 7 | **70.8** | **93.9** |
| 9 | 70.5 | 93.6 |

the fusion connection based on the small performance differences on both datasets. We choose the bi-direction fusion as our base fusion method.

### 4.2.6 Evaluation of A Number Of Segments

We evaluate the number of segments according to the default fusion method and architecture. One may assume that more segments result in better performance. However, as we discussed, more redundant temporal dynamics are introduced when increasing the number of segments. The performances based on different number of segments are shown in Table 4.4. It turns out that 7 segments performs best and 0.4% performance increases are observed on both datasets. The STFN with 9 segments underperforms compared with the one with with 7 segments. We verify our

Table 4.5: Performance comparison(%) of two-stream networks with ResNet-101 and Inception-V3 on HMDB51 and UCF101 (split1). Inception-V3 shows consistently better prediction accuracies over ResNet-101 on both appearance and motion networks.

| Dataset | Network | Appear. | Motion | Late Fusion |
|---|---|---|---|---|
| HMDB51 | ResNet-101 | 48.2 | 58.1 | 61.1 |
| | Inception-V3 | 51.2 | 59.2 | 62.7 |
| UCF101 | ResNet-101 | 83.5 | 86.0 | 91.8 |
| | Inception-V3 | 84.8 | 88.1 | 92.3 |

hypothesis with this experiments that sparse sampling is necessary to avoid redundant temporal dynamics over entire videos. For the best network, we determine the number of segments as 7.

### 4.2.7 Base Performance of Two-Stream Network

We compare the different ConvNet architectures for STFN. ResNet-101 [36] and Inception-V3 [83] networks are employed to train the two-stream networks. As shown in Table 4.5, the performance with Inception-V3 is better than ResNet-101 on both datasets. The performance gaps of the appearance and motion networks are 3.0%/1.1% on HMDB51 and 1.3%/2.1% on UCF101, respectively.

### 4.2.8 Comparison with the State-of-the-art

We compare STFN with the current state-ot-the-art methods in Table 4.6. We report the mean accuracy over three splits of the HMDB51 and UCF101. The first section of Table 4.6 consists of the hand-crafted features with different encoding methods. The second and third sections describe approaches using ConvNets but the methods in third section utilize additional modalities for the final prediction. STFN with the Inception-V3 achieves the best results: 72.1% on HMDB51 and 95.4% on UCF101. There is 0.9%/1.1% performance increase from STFN with ResNet-101 architecture. STFN with both networks shows the state-of-the-art performance. Comparing with baseline late fusion performance of two-stream networks, performance increases are observed as follows: 9.4%, 10.1% on HMDB51 and 3.1%, 2.5% on UCF101 with Inception-V3 and ResNet-101, respectively.

Our best results outperform TSN [208] by 1.0% on HMDB51 and 0.5% on UCF101 with same number of segments, 7. While TSN predicts scores with consensus operations and averages each score, STFN extracts temproal dynamic information and aggregates signals in feature level leading to better results. The results prove our method produces effective spatio-temporal features. DOVF [105] and TLE [34] show better results than STFN with ResNet-101 but are outperformed by STFN with Inception-V3. TLE [34] only outperforms our method with small margin, 0.2%, on UCF101 but the gap is reversed with additional hand-crafted feature score.

We combine our results with the hand-crafted MIFS[1] [104] features by averaging prediction scores. The performance gain on HMDB51, 3,0%, is larger than on UCF101, 1.6%. The combined

[1]The prediction scores of MIFS are downloaded from HERE.

performances, 75.1% on HMDB51 and 96.0% on UCF101, outperform all state-of-the-arts and even on par with [7, 176] which employ more prediction scores from additional modalities. Note that we observe similar performance boost with iDT [200] but choose MIFS since the prediction scores are available in public.

## 4.3 Conclusion

In this paper, we introduced the sptio-temporal fusion network (STFN), a network suitable for extracting temporal dynamics of features and learning spatio-temporal features by combining them. The spatio-temporal features are learned effectively with STFN via an end-to-end learning method. In the ablation studies, we show the best fusion methods and architecture and investigate the intuition behind each method. STFN enables appearance and motion dynamic features integrate inside of the networks in a highly abstract manner and overcomes the naive fusion strategy of late fusion. STFN is applicable to any sequencial data with two different modalities and effectively fuses them into highly discriminative feature that captures dynamic information over the entire sequence. The best result of STFN achieves the state-of-the-art performance, 75.1% on HMDB51 and 96.0% on UCF101. As future work, we consider scalability of our work with larger dataset and applying more than two modalities.

Table 4.6: Comparison with state-of-the-art methods on HMDB51 and UCF101. Mean accuracy over three splits. Numbers inside of parenthesis are classification accuracies with hand-crafted features. (i: iDT [200], H: HMG [41], M: MIFS [104])

| Methods | HMDB51 | UCF101 |
|---|---|---|
| iDT+FV [202] | 57.2 | 85.9 |
| iDT+HSV [140] | 61.1 | 87.9 |
| Two-stream [164] | 59.4 | 88.0 |
| Transformation [215] | 62.0 | 92.4 |
| KVM [237] | 63.3 | 93.1 |
| Two-Stream Fusion [47] | 65.4 (69.2 i) | 92.5 (93.5 i) |
| ST-ResNet [45] | 66.4 (70.3 i) | 93.4 (94.6 i) |
| ST-Multiplier [46] | 68.9 (72.2 i) | 94.2 (94.9 i) |
| ActionVLAD [60] | 66.9 (69.8 i) | 92.7 (93.6 i) |
| ST-Vector [26] | 69.5 (73.1 i+H) | 93.6 (94.3 i+H) |
| DOVF [105] | 71.7 (75.0 M) | 94.9 (95.3 M) |
| ST-Pyramid [217] | 68.9 | 94.6 |
| I3D [1] | 66.4 | 93.4 |
| CO2FI [116] | 69.0 (72.6 i) | 94.3 (95.2 i) |
| TLE [34] | 71.1 | **95.6** |
| TSN [208] | 71.0 | 94.9 |
| Four-Stream [7] | 72.5 (74.9 i) | 95.5 (96.0 i) |
| OFF [176] | 74.2 | 96.0 |
| STFN (ResNet-101) | 71.2 (73.3 M) | 94.3 (95.1 M) |
| STFN (Inception-V3) | **72.1** (**75.1** M) | **95.4** (**96.0** M) |

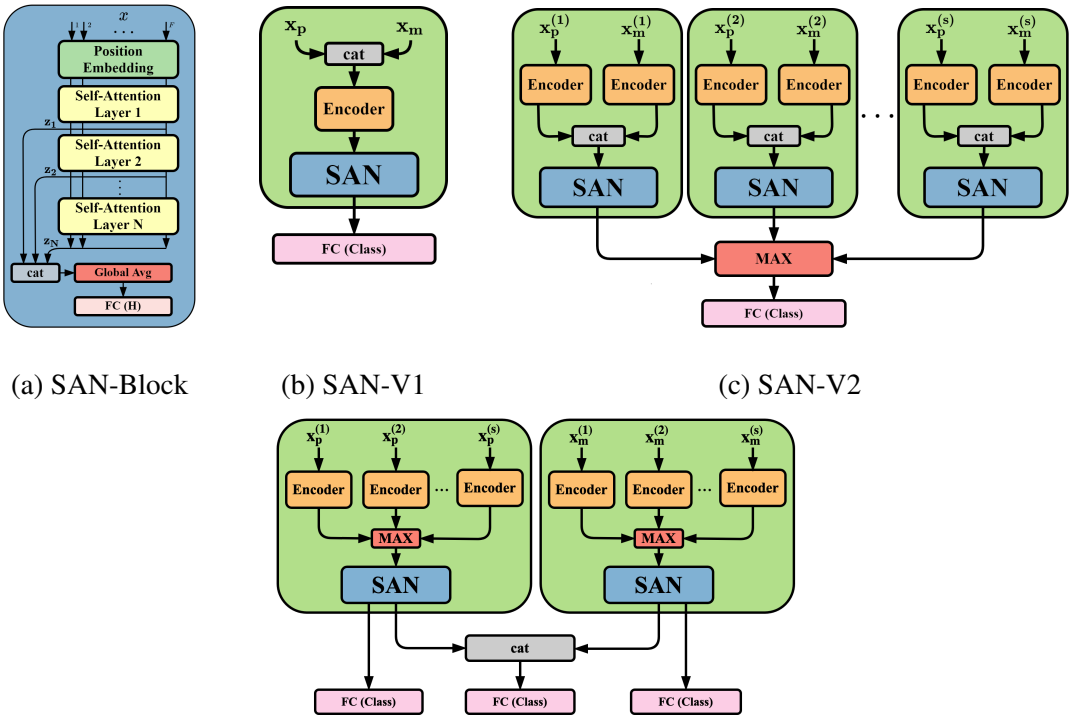
# CHAPTER 5
# SELF-ATTENTION NETWORK FOR SKELETON-BASED HUMAN ACTION RECOGNITION

## 5.1 Self-Attention Network

In this section, we briefly review the Self-attention network. Self-attention network [196] is a powerful method to compute correlation between arbitrary positions of a sequence input. An attention function consists of a query $\mathsf{A}_Q$, keys $\mathsf{A}_K$, and values $\mathsf{A}_V$ where query and keys have same vector dimension $\mathsf{d}_k$, and values and outputs have same size of dimension $\mathsf{d}_v$. The output is computed as a weighted sum of the values, and the weight assigned to each value is computed by scaled dot-product of query and keys. The vectors of query $\mathsf{A}_Q$, keys $\mathsf{A}_K$ and values $\mathsf{A}_V$ are packed in a matrix generating $\mathsf{Q}$, $\mathsf{K}$, and $\mathsf{V}$ matrices. Then the attention function is defined as

$$Attention(\mathsf{Q},\mathsf{K},\mathsf{V}) = softmax\left(\frac{\mathsf{Q}\mathsf{K}^T}{\sqrt{\mathsf{d}_k}}\right)\mathsf{V}, \tag{5.1}$$

where $\frac{1}{\sqrt{\mathsf{d}_k}}$ is a scaling factor.The equation computes scaled dot-product attention and the network computes the attention multiple times in parallel (multi-head) to extract different correlation information. The multi-head attention outputs are concatenated and transformed to the same vector dimension the input sequence. A residual connection is adopted to take the input and output of the multi-head self-attention layer and a layer normalization is applied to the summed output. A




(a) SAN-Block (b) SAN-V1 (c) SAN-V2

(d) SAN-V3

Figure 5.1: Different designs of Self-Attention Network architecture. (a) self-attention network block (SAN) computing pairwise correlated attentions; (b) baseline model with early fused input features; (c) model that learns movements of each person in a scene; (d) model that learn different modalities for available people in a scene.

fully-connected feed-forward network with a residual connection is applied to the normalized self-attention output. The entire network is illustrated as a self-attention layer in Fig. 5.1a and multiple layers are repeated to extract better representation.

## 5.2 Approach

In this section, we propose an effective model for skeleton-based action recognition, which is based on Self-Attention Network. The overall framework of the model is shown in Fig. 1.4. Primarily we have position and motion of joints. We can use raw position of the joints for figuring out the motion/velocity of the joints. Our SAN variants operate on encoded representations of position and motion sequences. We will be using simple non-linear projection (FCNN) and CNN based encoders for encoding the raw position and velocity sequences. First we will explain the data transformation from raw sequences of position and motion of the joints to encoded features. Once features are encoded, we will make use of three different SAN based architectures for effectively capturing the contextual information from the encoded features.

### 5.2.1 Raw Position and Motion Data

The raw skeleton position $\mathbf{x_p} \in \mathbb{R}^{F \times J' \times C}$ in a video clip is defined with the number of frames $F$, the number of joints per person $J$, and the coordinates of each joint $C$. There may be $S$ skeletons in a frame so the total number of joints is $J' = S \times J$. The position data can be depicted for each person as $\mathbf{x_p}^{(s)}$, where $s \in \{1, 2, \cdots, S\}$.

The motion or velocity data, $\mathbf{x_m} \in \mathbb{R}^{F \times J' \times C}$, can be explicitly retrieved by taking differences of each joint $J_j^t \in \mathbb{R}^C$, where $j \in \{1, \cdots, J\}$ and $t \in \{1, \cdots, F\}$, between consecutive frames:

$$\mathbf{x_m^t} = \left\{ J_1^{t+1} - J_1^t, J_2^{t+1} - J_2^t, \cdots, J_{\mathsf{J}}^{t+1} - J_{\mathsf{J}}^t \right\} \tag{5.2}$$

Similarly, the motion data for each person is represented as $\mathbf{x}_{\mathbf{m}}^{(s)}$.

### 5.2.2 Encoder

Our SAN variant models (Fig. 5.1) operate upon the encoded position $\mathbf{x}_{(\mathbf{p},\mathbf{enc})}$ and motion features $\mathbf{x}_{(\mathbf{m},\mathbf{enc})}$. In this section, we describe two methods to encode the raw position $\mathbf{x}_{\mathbf{p}}$ and motion data $\mathbf{x}_{\mathbf{m}}$.

#### 5.2.2.1 Non-Linear Encoder

A non-linear encoder simply uses a feed-forward neural network (FCNN) with a non-linear activation function for projecting the input vector to higher dimension. For example, when encoding for SAN-V1 (Fig. 5.1b) we perform early fusion of $\mathbf{x}_{\mathbf{p}}$ and $\mathbf{x}_{\mathbf{m}}$ to get $\mathbf{x} \in \mathbb{R}^{F \times 2J' \times C}$ and then use our non-linear encoder to get $\mathbf{x}_{(\mathbf{ff})} \in \mathbb{R}^{F \times 2J' \times C'}$. On the other hand, encoding for SAN-V2 (Fig. 5.1c) and SAN-V3 (Fig. 5.1d) individual skeletons are incorporated. In this case non-linear encoding is used to extend the skeleton joint position and motion tensor to $\mathbf{x}_{(\mathbf{p},\mathbf{ff})}^{(s)} \in \mathbb{R}^{F \times J \times C'}$, and $\mathbf{x}_{(\mathbf{m},\mathbf{ff})}^{(s)} \in \mathbb{R}^{F \times J \times C'}$, respectively.

#### 5.2.2.2 CNN Based Encoder

A CNN based encoder is employed for encoding low level features from raw joint position and motion data $\mathbf{x_p}$, $\mathbf{x_m}$, or $\mathbf{x_p}^{(s)}$, and $\mathbf{x_m}^{(s)}$. 2D convolutions can serve the purpose of extracting features from 3D tensors of raw skeleton data. Our encoder block consist of 4 convolutional layers as evident from Fig. 5.2. We will explain the general encoding scheme by keeping in view the encoding requirements for SAN-V1 architecture. As we mentioned earlier in 5.2.2.1, for SAN-V1, $\mathbf{x} \in \mathbb{R}^{F \times 2J' \times C}$ which is the output of early fusion of $\mathbf{x_p}$ and $\mathbf{x_m}$. First layer uses $1 \times 1 \times 64$ filters with stride 1. Output of the first layer are the extended coordinates in the form of $F \times J' \times 64$ tensor. Layer two operates with $3 \times 1 \times 32$ filters and stride 1, and outputs a tensor of shape $F \times J' \times 32$. Note that convolution window size for layer two is $3 \times 1$ because we are interested in extracting local contextual information over frames. Now, we transpose joints and cooridinates making the tensor of shape $F \times 32 \times J'$ in order to extract features from correlations of all joints over local frames. Third layer uses $3 \times 3 \times 32$ filters with stride 1 and max pooling with $1 \times 2$ pooling window is also applied. Output of third layer is a tensor with shape $F \times 16 \times 32$. Final convolution layer applies $3 \times 3 \times 64$ filters with stride 1. Similar to third layer, max pooling with a pooling window of $1 \times 2$ is also applied producing a $F \times 8 \times 64$ tensor. Last two CNN layers encode correlated local features from all joints of human body. For SAN-V2 (Fig. 5.1c) and SAN-V3 (Fig. 5.1d) we encode $\mathbf{x_p}^{(s)}$ and $\mathbf{x_m}^{(s)}$ for individual skeletons in the frames. Note that $F$ remains the same so feature representations for each frame are acquired with encoders.

### 5.2.3 SAN Variant Architecture

We investigate three SAN based network architectures as shown in Fig. 5.1 for skeleton based action recognition. These architectures employ the same SAN architecture as shown in Fig. 5.1a but operate upon varying combinations of encoded features, $\mathbf{x}_{(\mathbf{enc})}$, $\mathbf{x}_{(\mathbf{p},\mathbf{enc})}$, and $\mathbf{x}_{(\mathbf{m},\mathbf{enc})}$. We first discuss the SAN block used in the network in detail.

#### 5.2.3.1 Self-Attention Network

SAN block operates on encoded representations of position and motion information. The input to SAN block is $x \in \mathbb{R}^{F \times H}$, where $H$ is a feature representation per frame. The dimension of $H$ relys on the different encoders and model variants, and $H = 512 = 8 \times 64$ with the CNN encoder for SAN-V1. The first layer of the SAN block is a position embedding generating $p \in \mathbb{R}^{F \times H}$. Position embedding layer is used for providing a sense of order to the feature vectors. The ordering prior knowledge is helpful for each feature vector at each time to capture overall contextual cues from the input sequence. The output of the position embedding layer $y$ is an element-wise addition of the input sequence $x$ and the position embedding $p$.

Output of position embedding layer $y$ is fed to the first self-attention layer $\mathbf{z}_1$. Each SAN layer consumes the output of the previous SAN layer. Each self-attention layer computes pairwise attention probabilities and $K, Q$ and $V$ parameters described in Eq. 5.1 are learned. Each self-attention layer outputs $\mathbf{z}_i, i \in \{1, 2, \cdots, N\}$ where $N$ is the number of self-attention layers. We

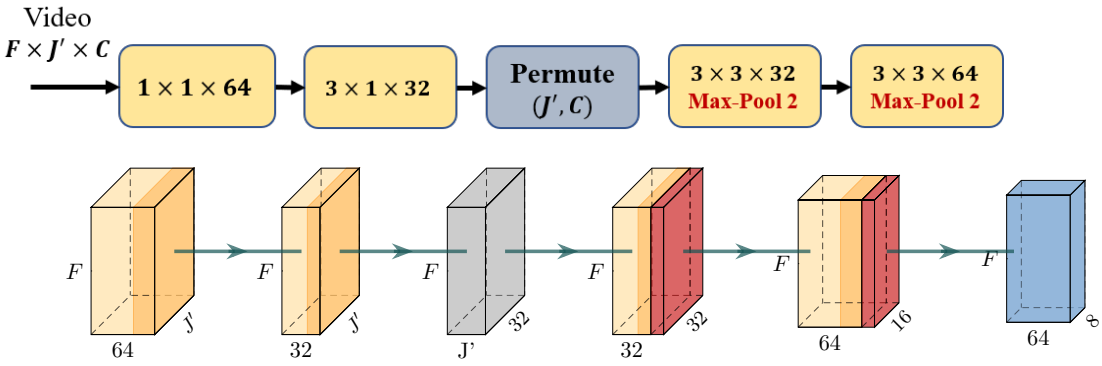



Figure 5.2: An input sequence of skeleton joints over frames, $F \times J' \times C$, is fed to the convolutional blocks and output tensor size of $F \times 8 \times 64$ is generated, which is denoted by . Each color denotes the following layers: convolutional layer; ReLU activation; and max-pooling layer.

concatenate the outputs from each SAN layer in order to gather all the attention probabilities as shown below

$$c = concat([\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N]) \tag{5.3}$$

$$o = \text{ReLU}(f_{lin}(f_{avg}(c))) \tag{5.4}$$

where *concat* layer concatenates $\mathbf{z}_i \in \mathbb{R}^{F \times H}$ along the vector axis creating a concatenated sequence $c \in \mathbb{R}^{F \times HN}$. Then, a global average layer $f_{avg}$ is applied to $c$ along the frame axis to obtain video-level features and a resulting dimension of the feature is $\mathbb{R}^{HN}$. Finally, a fully connected layer $f_{lin}$ with a non-linear activation, ReLU, projects the feature vector to the same input dimension $H$.

#### 5.2.3.2 SAN-V1

SAN-V1 (Fig. 5.1b) is a baseline network to understand how well the SAN block works for this task. It takes a concatenated input of position ($\mathbf{x_p}$) and motion ($\mathbf{x_m}$) data generating an input sequence $\mathbf{x} \in \mathbb{R}^{F \times 2J'C}$. The concatenation is to achieve feature-level early fusion. $\mathbf{x}$ requires encoding which is achieved using CNN encoder and non-linear encoder. The shape of the input sequence to the encoders is $\mathbb{R}^{F \times H}$ where $H = 2 \times J' \times C$. SAN block extracts latent local and global context information out of the input encoded sequences $\mathbf{x_{conv}}$ and $\mathbf{x_{ff}}$. Note that $J$ is the number of joints for one person, hence $J'$ represent the joints belonging to all the poeple in the frame. Zero paddings are applied in case that the number of valid people in a frame is less than a pre-defined maximum number of people. The output of the SAN block is fed to a classification layer which consists of a ReLU activation layer, a dropout layer, and a linear layer with softmax activation to predict probabilities for each class. The network is trained with cross-entropy loss.

#### 5.2.3.3 SAN-V2

SAN-V2 (Fig. 5.1c) is designed to extract contextual features with the SAN blocks for each subject (skeleton) in a scene. This network computes actions for each skeleton and takes the strongest signal from all available people in a video. Similar to SAN-V1, the encoded position and motion skeleton data for each person is concatenated respectively and the concatenated input sequences are fed to the corresponding SAN blocks. The input dimension for each SAN block is $\mathbb{R}^{F \times 2JC'}$ and

$\mathbb{R}^{F\times 2\times 512}$ with the non-linear and CNN encoder, respectively. SAN blocks share weights to learn a variety of movements from different people. SAN outputs can be merged with different operations such as element-wise max, mean or concatenation. According to our preliminary experiments, element-wise max works the best as it captures the strongest action signal among people who may not be available. The final classification layer is identical to the one in SAN-V1. Note that SAN-V2 leverages late fusion strategy and is scalable to arbitrary number of people.

#### 5.2.3.4 SAN-V3

Lastly, SAN-V3 (Fig. 5.1d) is designed to deal with different data modalities: position and velocity (or motion). The most prominent signals from all people are chosen by an element-wise max operation for each modality. The input dimension for the SAN block is $\mathbb{R}^{F\times JC'}$ and $\mathbb{R}^{F\times 512}$ for the non-linear and CNN encoder, respectively. The output of each SAN block is fed to separate classifiers and the concatenated signal from the SAN blocks is consumed by another classifier. This network is also scalable to any number of people in a scene. The training losses of the model are calculated by adding all cross entropy losses from each classifier.

### 5.2.4 Temporal Segment Self-Attention Network (TS-SAN)

The self-attention network can associate features in distance making it possible to capture long range information. However, as the feature representations for same action can vary with many

constraints (viewpoint change, different speed of action by different subjects, etc), the proposed network may not learn well. Thus, we leverage the temporal segment network [207] to train the network more effectively. As shown in Fig. 1.4, a video is divided into $K$ clips and one of the SAN variants in Fig. 5.1 is employed to learn temporal dynamics on each clip. Note that all layers share weights for different clips. Formally, given $K$ segments $S_1, S_2, \cdots, S_K$ of a video, the proposed network models a sequence of clips as follows:

$$TS-SAN(S_1, S_2, \cdots, S_K) = \mathscr{C}(\mathscr{F}(S_1; \mathbf{W}), \mathscr{F}(S_2; \mathbf{W}), \cdots, \mathscr{F}(S_K; \mathbf{W})). \tag{5.5}$$

where $\mathscr{F}$ denotes one of SAN-Variant models and $\mathbf{W}$ is its parameters. The predictions of each SAN model from each snippet are aggregated based on different function $\mathscr{C}$: element-wise max, and average.

## 5.3 Experiments

We perform extensive experiments to evaluate the effectiveness of our proposed Self-Attention frameworks on two large scale benchmark datasets: NTU RGB+D dataset [155], and Kinetics-skeleton dataset [84]. We analyze the performance of our variant models and visualize self-attention probabilities to understand its mechanism.

Table 5.1: Results of our method in comparison with state-of-the-art methods on NTU RGB+D with Cross-Subject(CS) and Cross-View(CV) benchmarks.

| Methods | CS | CV |
|---|---|---|
| H-RNN [39] (2015) | 59.1 | 64.0 |
| PA-LSTM [154] (2016) | 62.9 | 70.3 |
| TG ST-LSTM [118] (2016) | 69.2 | 77.7 |
| Two-stream RNN [203] (2017) | 71.3 | 79.5 |
| STA-LSTM [169] (2017) | 73.4 | 81.2 |
| Ensemble TS-LSTM [109] (2017) | 74.6 | 81.3 |
| VA-LSTM [231] (2017) | 79.4 | 87.6 |
| ST-GCN [222] (2018) | 81.5 | 88.3 |
| DPRL [183] (2018) | 83.5 | 89.8 |
| HCN [110] (2018) | 86.5 | 91.9 |
| SR-TSL [163] (2018) | 84.8 | 92.4 |
| TS-SAN (Ours) | **87.2** | **92.7** |

### 5.3.1 Datasets

#### 5.3.1.1 NTU RGB+D

NTU RGB+D is the current largest action recognition dataset with joints annotations that are collected by Microsoft Kinect v2. It has 56,880 video samples and contains 60 action classes in total. These actions are performed by 40 distinct subjects. It is recorded with three cameras simultaneously in different horizontal views. The joints annotations consist of 3D locations of 25 major body joints. [155] defines two standard evaluation protocols for this dataset: Cross-Subject (CS) and Cross-View (CV). For Cross-Subject evaluation, the 40 subjects are split into training and testing groups. Each group consists of 20 subjects. The numbers of training and testing samples are 40,320 and 16,560, respectively. For Cross-View evaluation, all the samples of cameras 2 and 3 are used for training while the samples of camera 1 are used for testing. The numbers of training and testing samples are 37,920 and 18,960, respectively.

#### 5.3.1.2 Kinetics

Kinetics [84] contains about 266,000 video clips retrieved from YouTube and covers 400 classes. Since no skeleton annotation is provided, the skeleton is estimated by an OpenPose toolbox [11] from the resized videos of $340\times256$ resolution. The toolbox estimates 2D coordinates $(x,y)$ of 18 human joints and confidence scores $c$ for each joint. Each joint is represented as $(x,y,c)$ and

2 people are selected at most for each frame based on the highest average joint confidence score. The total number of frames for all clips is fixed to 300 by repeating the sequence from the start. We employ the released skeleton dataset to train our model and report the performance of top-1 and top-5 accuracies as introduced in [222]. The numbers of training and validation samples are around 246,000 and 20,000, respectively.

### 5.3.2 Implementation Details

We resize the sequence length to a fixed number of **F**=32/64 (NTU/Kinetics) with bilinear interpolation along the frame dimension. We use $K$=3 of temporal segments and 32 frames are sampled from each clip. The numbers of self-attention layers and multi-heads used for NTU RGB+D and Kinetics datasets are 4, 8 and 8, 8, respectively.

To alleviate the problem of overfitting, we append dropout with a probability of 0.5 before the last prediction layer and after the last convolution layer. For the self-attention network, a 0.2 ratio of dropout is utilized. We employ a data augmentation scheme by randomly cropping sequences with a ratio of uniform distribution between [0.5, 1] for training. We center crop sequence with a ratio of 0.9 when testing. The learning rate is initialized with $1e^{-4}$ and reduced by half in case no improvement of accuracy is observed for 5 epochs. Adam optimizer [91] is applied with weight decay of $5e^{-5}$. The model is trained for 200/100 (NTU/Kinetics) epochs with a batch size of 64.

Table 5.2: Results of our method in comparison with state-of-the-art methods on Kinetics.

| Methods | Top-1 | Top-5 |
|---|---|---|
| Feature Enc. [50] (2015) | 14.9 | 25.8 |
| Deep LSTM [154] (2016) | 16.4 | 35.3 |
| Temporal Conv [89] (2017) | 20.3 | 40.0 |
| ST-GCN [222] (2018) | 30.7 | 52.8 |
| TS-SAN (Ours) | **35.1** | **55.7** |

### 5.3.3 Comparison to State of the art

We compare the performance of the proposed method to the state-of-the-art methods on NTU RGB+D and Kinetics datasets as shown in Table 5.1 and Table 5.2. The compared methods are based on CNN, RNN (or LSTM), and graph structure and our method consistently outperform state-of-the-art approaches. This demonstrates the effectiveness of our proposed model for the skeleton-based action recognition task.

As shown in Table 5.1, our proposed model achieves the best performance with 87.2% with CS and 92.7% with CV. Our model and [169] have common in a sense that attention mechanism is used. By comparing with STA-LSTM [169], our model performs 13.8% with CS and 11.5% with CV. Our model encodes the raw skeleton data with CNNs similar to HCN [110] but outperforms

by 0.7% with CS and 0.8% with CV. Comparing our model with SR-TSL [163] which is one of the best-performed methods, the performance gaps are 2.4% with CS and 0.3% with CV.

On the Kinetics dataset, we compare with four methods which are based on handcraft features, LSTM, temporal convolution, and graph-based convolution. As shown in Table 5.2, our method attains the best performance with a significant margin. The proposed method outperforms by 4.4% on top-1 and 2.9% on top-5 accuracies. We observe that CNN based methods [110, 163, 222, 89] are superior to LSTM based methods [231, 109, 154] based on both Table 5.1 and Table 5.2, and our model outperforms the CNN based methods.

### 5.3.4 Ablation Study

We analyze the proposed network by comparing it with baseline models. We compare SAN variants with hyperparameter options for encoders, self-attention network, and temporal segment network. Each experiment is evaluated on the NTU RGB+D dataset.

#### 5.3.4.1 Effect of SAN Variants with Different Encoders

Table 5.3 shows the results with different SAN variants and different inputs to them. The SAN-V2 model performs the best and the SAN-V1 model the worst. The gap between the SAN-V2 model and the SAN-V3 model is minimal. We observe that the CNN encoder boosts the performance accuracy by up to 7.3% for SAN-V3. It shows that the CNN encoder effectively generates rich

Table 5.3: The comparison results of SAN variants shown in Fig. 5.1 with different encoder inputs on NTU dataset (%).

| Methods | CS | CV |
|---|---|---|
| SAN-V1 + FF | 75.4 | 79.8 |
| SAN-V1 + CNN | 80.1 | 86.2 |
| SAN-V2 + FF | 80.3 | 85.2 |
| SAN-V2 + CNN | 85.9 | 91.7 |
| SAN-V3 + FF | 78.6 | 84.1 |
| SAN-V3 + CNN | 85.5 | 91.4 |

Table 5.4: The comparison results of effectiveness of temporal segment on NTU dataset (%).

| Methods | CS | CV |
|---|---|---|
| SAN-V2 (seq=96) | 86.1 | 92.0 |
| SAN-V3 (seq=96) | 85.9 | 91.7 |
| TS (seg=3) + SAN-V2 (seq=32) | 87.2 | 92.7 |
| TS (seg=3) + SAN-V3 (seq=32) | 86.8 | 92.4 |

feature representations for the SAN models and plays a significant role in the network. From the observation that SAN-V2 slightly outperforms SAN-V3, we conclude two facts: late fusion performs better than early fusion; and sharing weights of SAN blocks resulting in better trained models.

#### 5.3.4.2 Effect of Temporal Segment

The self-attention network is suitable for connecting both short and long-range features and is capable of capturing higher-level context from all correlations. We compare the TS-SAN and SAN variants to see how they perform differently if two networks have the same sequence length. As shown in Table 5.4, TS-SAN outperforms. This proves that our design goal to make use of the temporal segment is correct. However, the SAN variants without the temporal segment network have an advantage of having less parameters with a small sacrifice of performance. Although TS-SAN models outperform, we observe that the SAN variants perform well for long-range input sequences, $F$=96.

#### 5.3.4.3 Effect of Consensus Function

We consider element-wise operations for the consensus function to compute the final prediction. Two operations are valid: element-wise average, element-wise maximum. Table 5.5 shows the performances of TS-SAN-V2 and TS-SAN-V3 with the above operations. The element-wise av-

Table 5.5: The comparison results of different aggregation methods for TS network on NTU dataset (%).

| Methods | CS | CV |
|---|---|---|
| TS(Avg) + SAN-V2 | 87.2 | 92.7 |
| TS(Max) + SAN-V2 | 86.1 | 91.9 |
| TS(Avg) + SAN-V3 | 86.8 | 92.4 |
| TS(Max) + SAN-V3 | 85.9 | 91.1 |

Table 5.6: The comparison results of the number of attention layers and multi-heads on NTU dataset (%).

| Methods | CS | CV |
|---|---|---|
| TS + SAN-V2 (L2H2) | 86.7 | 92.1 |
| TS + SAN-V2 (L4H4) | 86.9 | 92.5 |
| TS + SAN-V2 (L4H8) | 87.2 | 92.7 |
| TS + SAN-V2 (L8H8) | 87.0 | 92.4 |

erage consensus function outperforms the element-wise max operation in both SAN variants. The TS-SAN model with the element-wise max operation is outperformed by the SAN model without

the temporal segment as shown in Table 5.4. We conjecture that since the self-attention output signals are based on weighted average computation, it makes more sense to use the element-wise average aggregation function for the collected outputs from each snippet. By doing so, the video level self-attention can be computed properly leading to the best performance.

#### 5.3.4.4 Effect of Number of Layers and Mutli-Heads in SAN Block

We compare TS-SAN-V2 model with different number of layers and multi-heads. The results are shown in Table 5.6. By comparing the row 2 and 3, we observe that the number of heads affect the performance marginally. From the results of the row 3 and 4, we also observe that the network underperform if it contains too many paramerters. On the contrary, the network also underperforms when the number of parameters are not enough (row 1). According to the results, we argue that the proposed model requires a proper number of layers and heads for a cetrain dataset to perform the best.

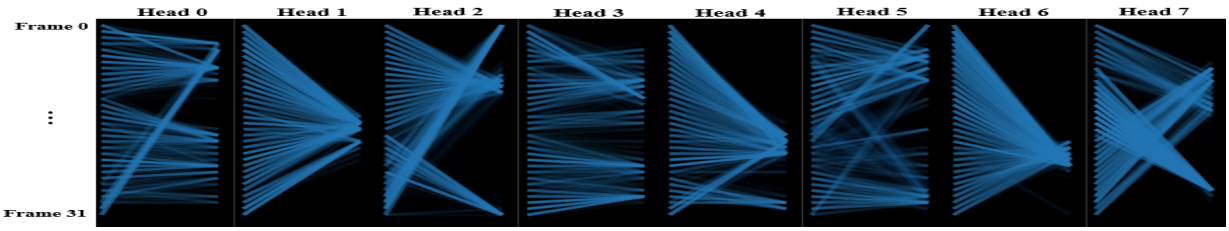
### 5.3.5 Visualization of Self-Attention Layer Response

The self-attention network determines where each frame correlates to other frames. We visualize the self-attention response from the last self-attention layer with a visualization tool [198] to understand how each frame is correlated for a certain action video. As shown in Fig. 5.3, the vertical

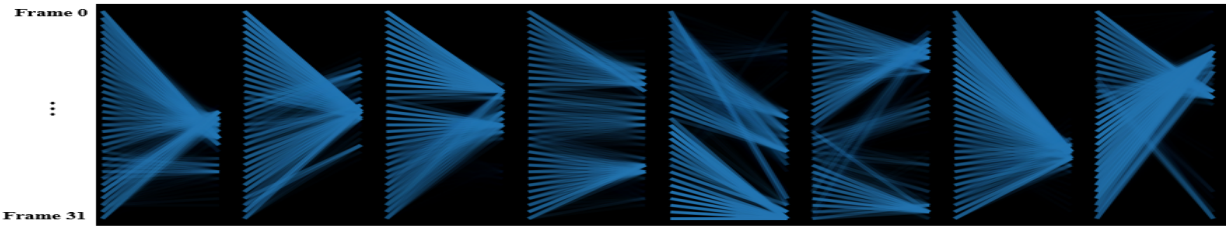
axis shows the sampled 32 frames. Self-attention responses for eight multi-heads are displayed and each column shows the coarse shape of the attention pattern between two frames.

The model used for this visualization attains four layers and eight heads, and takes 32 sampled frames as the input sequence. No temporal segment network is used to train the network. The self-attention probabilities are calculated by the equ. 5.1 in the self-attention layer described in Fig. 5.1a. For example, from Fig. 5.3a, one of the strongest correlation in the third head can be found from a connection between frame 31 to frame 0 (a line across from bottom left to top right). From the above example, we can check the long range correlation is achieved and the proposed method captures a variety of correlations in both short and long distance.

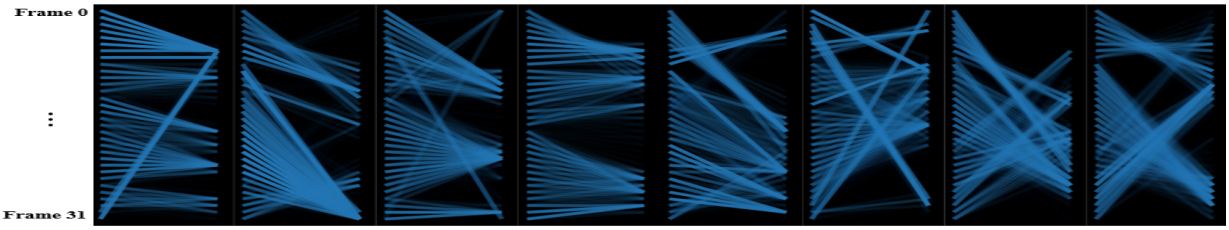
We observe that the overall self-attention response patterns of the same action class (‘put on jacket’) resembles each other as shown in Fig. 5.3a and Fig. 5.3b. The repsonses of head 1 and head 6 from two videos especially shows similar pattern. Although two videos are taken by different subjects, duration, and views, we can see that the self-attention catches a certain latent similarity. Comparing Fig. 5.3a and Fig. 5.3b with Fig. 5.3c, there is not much similar response pattern between them due to different action classes (‘put on jacket’ vs ‘reading’). We also learn that the proposed model is robust to subtle motion or speed of action changes from difference subjects or even views.




(a) 'Put on jacket' action with subject 1




(b) 'Put on jacket' action with subject 2




(c) 'Reading' action with subject 1

Figure 5.3: Self-attention probabilities from the last self-attention layer for three test videos on NTU RGB+D are visualized. The brighter color denotes the higher probability or the stronger connection.

## 5.4 Conclusion

In this paper, we propose three novel SAN variations in order to extract high-level context from short and long-range self-attentions. Our proposed architectures significantly outperform state-

of-the-art methods. CNN employed in our model is effective to extract feature representations for the input sequence of the self-attention network. SAN can capture the temporal correlations regardless of distance, making it possible to obtain high-level context information from both short and long-range self-attentions. We also propose an effective integration of SAN and TSN which results in observable performance boost. We perform extensive experiments on two large scale datasets, NTU RGB+D and Kinetics-skeleton, and verify the effectiveness of our proposed models for the skeleton-based action recognition task. In the future, we will apply our model to video-based recognition tasks with key point annotations, such as facial expression recognition. We will also explore different methods to extract effective feature representations for the input sequence of SAN.

# CHAPTER 6
# IMPROVING THE SIMILARITY MEASURE OF DETERMINANTAL POINT PROCESSES FOR EXTRACTIVE MULTI-DOCUMENT SUMMARIZATION

## 6.1 The DPP Framework

Let $\mathscr{Y} = \{1, 2, \cdots, \mathsf{N}\}$ be a ground set containing $\mathsf{N}$ items, corresponding to all sentences of the source documents. Our goal is to identify a subset of items $Y \subseteq \mathscr{Y}$ that forms an extractive summary of the document set. A determinantal point process (DPP; Kulesza and Taskar, 2012) defines a probability measure over all subsets of $\mathscr{Y}$ s.t.

$$\mathscr{P}(Y; L) = \frac{\det(L_Y)}{\det(L+I)}, \tag{6.1}$$

$$\sum_{Y \subseteq \mathscr{Y}} \det(L_Y) = \det(L+I), \tag{6.2}$$

where $\det(\cdot)$ is the determinant of a matrix; $I$ is the identity matrix; $L \in \mathbb{R}^{\mathsf{N} \times \mathsf{N}}$ is a positive semidefinite matrix, known as the $L$-ensemble; $L_{ij}$ measures the correlation between sentences $i$ and $j$; and $L_Y$ is a submatrix of $L$ containing only entries indexed by elements of $Y$. Finally, the probability of an extractive summary $Y \subseteq \mathscr{Y}$ is proportional to the determinant of the matrix $L_Y$ (Eq. (6.1)).

Kulesza and Taskar [101] provide a decomposition of the $L$-ensemble matrix: $L_{ij} = q_i \cdot S_{ij} \cdot q_j$ where $q_i \in \mathbb{R}^+$ is a positive real number indicating the *quality* of a sentence; and $S_{ij}$ is a measure of *similarity* between sentences $i$ and $j$. This formulation separately models the sentence quality and pairwise similarity before combining them into a unified model. Let $Y = \{i, j\}$ be a summary containing only two sentences $i$ and $j$, its probability $\mathscr{P}(Y;L)$ can be computed as

$$\begin{aligned}\mathscr{P}(Y = \{i,j\};L) &\propto \det(L_Y) \\ &= \begin{vmatrix} q_i S_{ii} q_i & q_i S_{ij} q_j \\ q_j S_{ji} q_i & q_j S_{jj} q_j \end{vmatrix} \\ &= q_i^2 \cdot q_j^2 \cdot (1 - S_{ij}^2). \end{aligned} \tag{6.3}$$

Eq. (6.3) indicates that, if sentence $i$ is of high quality, denoted by $q_i$, then any summary containing it will have high probability. If two sentences $i$ and $j$ are similar to each other, denoted by $S_{ij}$, then any summary containing both sentences will have low probability. The summary $Y$ achieving the highest probability thus should contain a set of high-quality sentences while maintaining high diversity among the selected sentences (via pairwise repulsion). $\det(L_Y)$ also has a particular geometric interpretation as the squared volume of the space spanned by sentence vectors $i$ and $j$, where the quality measure indicates the length of the vector and the similarity indicates the angle between two vectors (Figure 6.1).

We adopt a feature-based approach to compute sentence quality: $q_i = \exp(\theta^\top \mathbf{x}_i)$. In particular, $\mathbf{x}_i$ is a feature vector for sentence $i$ and $\theta$ are the feature weights to be learned during training. Kulesza and Taskar [100] define sentence similarity as $S_{i,j} = \phi_i^\top \phi_j$, where $\|\phi_i\|_2 = 1$ $(\forall i)$ is a sen-

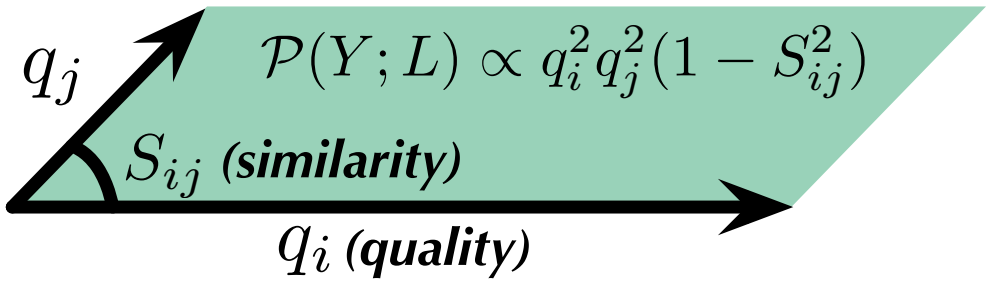



Figure 6.1: The DPP model specifies the probability of a summary $\mathscr{P}(Y = \{i, j\}; L)$ to be proportional to the squared volume of the space spanned by sentence vectors $i$ and $j$.

tence TF-IDF vector. The model parameters $\theta$ are optimized by maximizing the log-likelihood of training data (Eq. (6.4)) and this objective can be optimized efficiently with subgradient descent.[1]

$$\theta = \arg\max_{\theta} \sum_{m=1}^{M} \log \mathscr{P}(\hat{Y}^{(m)}; L(\mathscr{Y}^{(m)}; \theta)) \tag{6.4}$$

During training, we create the ground-truth extractive summary ($\hat{Y}$) for a document set based on human reference summaries (abstracts) using the following procedure. At each iteration we select a source sentence sharing the longest common subsequence with the human reference summaries; the shared words are then removed from human summaries to avoid duplicates in future selection. Similar methods are exploited by Nallapati et al. [127] and Narayan et al. [130] to create ground-truth extractive summaries. At test time, we perform inference using the learned DPP model to obtain a system summary ($Y$). We implement a greedy method (Kulesza and Taskar, 2012) to

[1]The sentence features include the length and position of a sentence, the cosine similarity between sentence and document TF-IDF vectors [100]. We refrain from using sophisticated features to avoid model overfitting.

iteratively add a sentence to the summary so that $\mathscr{P}(Y;L)$ yields the highest probability (Eq. (6.1)), until a summary length limit is reached.

For the DPP framework to be successful, the sentence similarity measure ($S_{ij}$) has to accurately capture if any two sentences contain redundant information. This is especially important for multi-document summarization as redundancy is ubiquitous in source documents. The source descriptions frequently contain redundant yet lexically diverse expressions such as sentential paraphrases where people write about the same event using distinct styles [73]. Without accurately modelling sentence similarity, redundant content can make their way into the summary and further prevent useful information from being included given the summary length limit. Existing cosine similarity measure between sentence TF-IDF vectors can be incompetent in modeling semantic relatedness. In the following section we exploit the recently introduced capsule networks [69] to measure pairwise sentence similarity; it considers if two sentences share any words in common and more importantly the semantic closeness of sentence descriptions.

## 6.2 An Improved Similarity Measure

Our goal is to develop an advanced similarity measure for pairs of sentences such that semantically similar sentences can receive high scores despite that they have very few words in common. E.g., "*Snowstorm slams eastern US on Friday*" and "*A strong wintry storm was dumping snow in eastern US after creating traffic havoc that claimed at least eight lives*" have only two words in common. Nonetheless, they contain redundant information and cannot both be included in the summary.

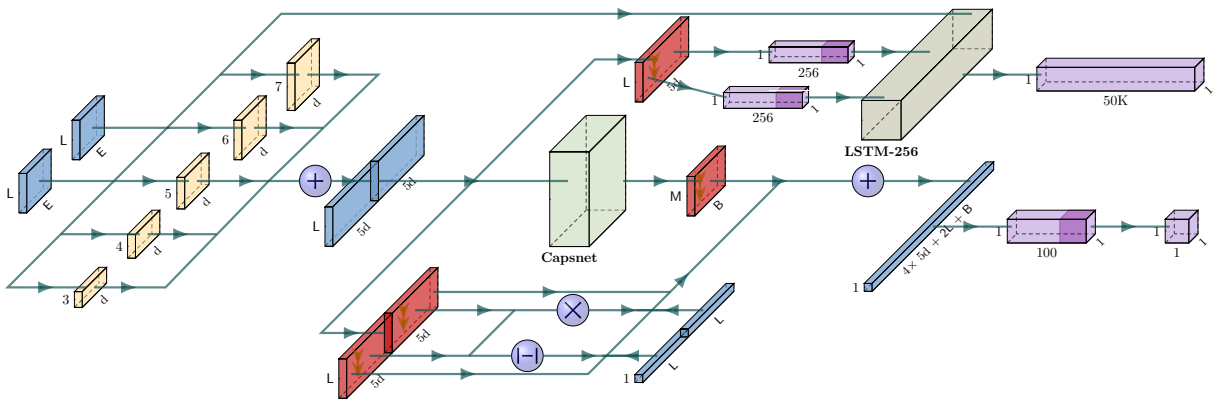



Figure 6.2: The system architecture utilizing CapsNet for predicting sentence similarity. denotes the inputs and intermediate outputs; the convolutional layer; max-pooling layer; fully-connected layer; and ReLU activation.

Let $\{\mathbf{x}^{\mathsf{a}}, \mathbf{x}^{\mathsf{b}}\} \in \mathbb{R}^{\mathsf{E} \times \mathsf{L}}$ denote two sentences $\mathsf{a}$ and $\mathsf{b}$. Each consists of a sequence of word embeddings, where $\mathsf{E}$ is the embedding size and $\mathsf{L}$ is the sentence length with zero-padding to the right for shorter sentences. A convolutional layer with multiple filter sizes is first applied to each sentence to extract local features (Eq. (6.5)), where $\mathbf{x}^{\mathsf{a}}_{i:i+k-1} \in \mathbb{R}^{k\mathsf{E}}$ denotes a flattened embedding for position $i$ with a filter size $k$, and $\mathbf{u}^{\mathsf{a}}_{i,k} \in \mathbb{R}^{d}$ is the resulting local feature for position $i$; $f$ is a nonlinear activation function (e.g., ReLU); $\{\mathbf{W}^{u}, \mathbf{b}^{u}\}$ are model parameters.

$$\mathbf{u}^{\mathsf{a}}_{i,k} = f(\mathbf{W}^{u}\mathbf{x}^{\mathsf{a}}_{i:i+k-1} + \mathbf{b}^{u}) \tag{6.5}$$

We use $\mathbf{u}^{\mathsf{a}}_{i} \in \mathbb{R}^{\mathsf{D}}$ to denote the concatenation of local features generated using various filter sizes. Following Kim et al. [87], we employ filter sizes $k \in \{3,4,5,6,7\}$ with an equal number of filters ($d$) for each size ($\mathsf{D} = 5d$). After applying max-pooling to local features of all positions, we obtain a representation $\mathbf{u}^{\mathsf{a}} = \text{max-pooling}(\mathbf{u}^{\mathsf{a}}_{i}) \in \mathbb{R}^{\mathsf{D}}$ for sentence $\mathsf{a}$; and similarly we

obtain $\mathbf{u}^{\mathsf{b}} \in \mathbb{R}^{\mathsf{D}}$ for sentence $\mathsf{b}$. It is not uncommon for state-of-the-art sentence similarity classifiers [16] to concatenate the two sentence vectors, their absolute difference and element-wise product $[\mathbf{u}^{\mathsf{a}}; \mathbf{u}^{\mathsf{b}}; |\mathbf{u}^{\mathsf{a}} - \mathbf{u}^{\mathsf{b}}|; \mathbf{u}^{\mathsf{a}} \circ \mathbf{u}^{\mathsf{b}}]$, and feed this representation to a fully connected layer to predict if two sentences are similar.

Nevertheless, we conjecture that such representation may be insufficient to fully characterize the relationship between components of the sentences in order to model sentence similarity. For example, the term "*snowstorm*" in sentence $\mathsf{a}$ is semantically related to "*wintry storm*" and "*dumping snow*" in sentence $\mathsf{b}$; this low-level interaction indicates that the two sentences contain redundant information and it cannot be captured by the above model. Importantly, the capsule networks proposed by Hinton et al. [69] are designed to characterize the spatial and orientational relationships between low-level components. We thus seek to exploit CapsNet to strengthen the capability of our system for identifying redundant sentences.

Let $\{\mathbf{u}_i^{\mathsf{a}}, \mathbf{u}_i^{\mathsf{b}}\}_{i=1}^{\mathsf{L}} \in \mathbb{R}^{\mathsf{D}}$ be low-level representations (i.e., capsules). We seek to transform them to high-level capsules $\{\mathbf{v}_j\}_{j=1}^{\mathsf{M}} \in \mathbb{R}^{\mathsf{B}}$ that characterize the interaction between low-level components. Each low-level capsule $\mathbf{u}_i \in \mathbb{R}^{\mathsf{D}}$ is multiplied by a linear transformation matrix to dedicate a portion of it, denoted by $\hat{\mathbf{u}}_{j|i} \in \mathbb{R}^{\mathsf{B}}$, to the construction of a high-level capsule $j$ (Eq. (6.6)); where $\{\mathbf{W}_{ij}^{v}\} \in \mathbb{R}^{\mathsf{D} \times \mathsf{B}}$ are model parameters. To reduce parameters and prevent overfitting, we further encourage sharing parameters over all low-level capsules, yielding $\mathbf{W}_{1j}^{v} = \mathbf{W}_{2j}^{v} = \cdots$, and the same parameter sharing is described in [234]. By computing the weighted sum of $\hat{\mathbf{u}}_{j|i}$, whose weights $c_{ij}$ indicate the strength of interaction between a low-level capsule $i$ and a high-level capsule $j$, we obtain an (unnormalized) capsule (Eq. (6.7)); we then apply a nonlinear squash function $g(\cdot)$ to normalize

the length the vector to be less than 1, yielding $\mathbf{v}_j \in \mathbb{R}^{\mathsf{B}}$.

$$\hat{\mathbf{u}}_{j|i} = \mathbf{W}^{v}_{ij}\mathbf{u}_i \tag{6.6}$$

$$\mathbf{v}_j = g\big(\sum_i c_{ij}\hat{\mathbf{u}}_{j|i}\big) \tag{6.7}$$

Routing [149, 233] aims to adjust the interaction weights ($c_{ij}$) using an iterative, EM-like method. Initially, we set $\{b_{ij}\}$ to be zero for all $i$ and $j$. Per Eq. (6.8), $\mathbf{c}_i$ becomes a uniform distribution indicating a low-level capsule $i$ contributes equally to all its upper level capsules. After computing $\hat{\mathbf{u}}_{j|i}$ and $\mathbf{v}_j$ using Eq. (6.6-6.7), the weights $b_{ij}$ are updated according to the strength of interaction (Eq. (6.9)). If $\hat{\mathbf{u}}_{j|i}$ agrees with a capsule $\mathbf{v}_j$, their interaction weight will be increased, and decreased otherwise. This process is repeated for $r$ iterations to stabilize $c_{ij}$.

$$\mathbf{c}_i \leftarrow \text{softmax}(\mathbf{b}_i) \tag{6.8}$$

$$b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}\mathbf{v}_j \tag{6.9}$$

The high-level capsules $\{\mathbf{v}_j\}_{j=1}^{\mathsf{M}}$ effectively encode spatial and orientational relationships of low-level capsules. To identify the most prominent interactions, we apply max-pooling to all high-level capsules to produce $\mathbf{v} = \text{max-pooling}_j(\mathbf{v}_j) \in \mathbb{R}^{\mathsf{B}}$. This representation $\mathbf{v}$, aimed to encode interactions between sentences $\mathsf{a}$ and $\mathsf{b}$, is concatenated with $[\mathbf{u}^{\mathsf{a}}; \mathbf{u}^{\mathsf{b}}; |\mathbf{u}^{\mathsf{a}} - \mathbf{u}^{\mathsf{b}}|; \mathbf{u}^{\mathsf{a}} \circ \mathbf{u}^{\mathsf{b}}]$ and binary vectors $[\mathbf{z}^{\mathsf{a}}; \mathbf{z}^{\mathsf{b}}]$ that indicate if any word in sentence $\mathsf{a}$ appears in sentence $\mathsf{b}$ and vice versa; they are used as input to a fully connected layer to predict if a pair of sentences contain redundant information. Our loss function contains two components, including a binary cross-entropy loss indicating whether the prediction is correct or not, and a reconstruction loss for reconstructing a sentence $\mathsf{a}$ conditioned on $\mathbf{u}^{\mathsf{a}}$ by predicting one word at a time using a recurrent neural network,

and similarly for sentence b. A hyperparameter $\lambda$ is used to balance contributions from both sides. In Figure 6.2 we present an overview of the system architecture, and hyper-parameters are described in the supplementary.

## 6.3 Datasets

To our best knowledge, there is no dataset focusing on determining if two sentences contain redundant information. It is a nontrivial task in the context of multi-document summarization. Further, we argue that the task should be distinguished from other semantic similarity tasks: semantic textual similarity (STS; Cer et al., 2017) assesses to what degree two sentences are semantically equivalent to each other; natural language inference (NLI; Bowman et al., 2015) determines if one sentence ("hypothesis") can be semantically inferred from the other sentence ("premise"). Nonetheless, redundant sentences found in a set of source documents discussing a particular topic are not necessarily semantically equivalent or express an entailment relationship. We compare different datasets in §6.4.

**Sentence redundancy dataset** A novel dataset containing over 2 million sentence pairs is introduced in this paper for sentence redundancy prediction. We hypothesize that it is likely for a summary sentence and its most similar source sentence to contain redundant information. Because humans create summaries using generalization, paraphrasing, and other high-level text operations, a summary sentence and its source sentence can be semantically similar, yet contain diverse expressions. Fortunately, such source/summary sentence pairs can be conveniently derived from

single-document summarization data. We analyze the CNN/Daily Mail dataset [67] that contains a massive collection of single news articles and their human-written summaries. For each summary sentence, we identify its most similar source sentence by calculating the averaged R-1, R-2, and R-L F-scores [114] between a source and summary sentences. We consider a summary sentence to have no match if the score is lower than a threshold. We obtain negative examples by randomly sampling two sentences from a news article. In total, our training / dev / test sets contain 2,084,798 / 105,936 / 86,144 sentence pairs and we make the dataset available to advance research on sentence redundancy.

**Summarization datasets** We evaluate our DPP-based system on benchmark multi-document summarization datasets. The task is to create a succinct summary with up to 100 words from a cluster of 10 news articles discussing a single topic. The DUC and TAC datasets [137, 28] have been used in previous summarization competitions. In this paper we use DUC-03/04 and TAC-08/09/10/11 datasets that contain 60/50/48/44/46/44 document clusters respectively. Four human reference summaries have been created for each document cluster by NIST assessors. Any system summaries are evaluated against human reference summaries using the ROUGE software [114][2], where R-1, -2, and -SU4 respectively measure the overlap of unigrams, bigrams, unigrams and skip bigrams with a maximum distance of 4 words. We report results on DUC-04 (trained on DUC-03) and TAC-11 (trained on TAC-08/09/10) that are often used as standard test sets [72].

[2] w/ options -n 2 -m -w 1.2 -c 95 -r 1000 -l 100

Table 6.1: ROUGE results on DUC-04. † indicates our reimplementation of Kulesza and Taskar [100].

| | DUC-04 | | |
|---|---|---|---|
| **System** | **R-1** | **R-2** | **R-SU4** |
| Opinosis [53] | 27.07 | 5.03 | 8.63 |
| Extract+Rewrite [168] | 28.90 | 5.33 | 8.76 |
| Pointer-Gen [152] | 31.43 | 6.03 | 10.01 |
| SumBasic [194] | 29.48 | 4.25 | 8.64 |
| KLSumm (Haghighi et al., 2009) | 31.04 | 6.03 | 10.23 |
| LexRank [42] | 34.44 | 7.11 | 11.19 |
| Centroid [72] | 35.49 | 7.80 | 12.02 |
| ICSISumm [58] | 37.31 | 9.36 | 13.12 |
| DPP [100]† | 38.10 | 9.14 | 13.40 |
| DPP-Capsnet (this work) | 38.25 | 9.22 | 13.40 |
| DPP-Combined (this work) | **39.35** | **10.14** | **14.15** |

## 6.4 Experimental Results

In this section we discuss results that we obtained for multi-document summarization and determining redundancy between sentences.

### 6.4.1 Summarization Results

We compare our system with a number of strong summarization baselines (Table 6.1 and 6.2). In particular, *SumBasic* [194] is an extractive approach assuming words occurring frequently in a

Table 6.2: ROUGE results on the TAC-11 dataset.

| System | TAC-11 R-1 | TAC-11 R-2 | TAC-11 R-SU4 |
|---|---|---|---|
| Opinosis [53] | 25.15 | 5.12 | 8.12 |
| Extract+Rewrite [168] | 29.07 | 6.11 | 9.20 |
| Pointer-Gen [152] | 31.44 | 6.40 | 10.20 |
| SumBasic [194] | 31.58 | 6.06 | 10.06 |
| KLSumm (Haghighi et al., 2009) | 31.23 | 7.07 | 10.56 |
| LexRank [42] | 33.10 | 7.50 | 11.13 |
| DPP [100]† | 36.95 | 9.83 | 13.57 |
| DPP-Capsnet (this work) | 36.61 | 9.30 | 13.09 |
| DPP-Combined (this work) | **37.30** | **10.13** | **13.78** |

document cluster are more likely to be included in the summary; *KL-Sum* [64] is a greedy approach adding a sentence to the summary to minimize KL divergence; and *LexRank* [42] is a graph-based approach computing sentence importance based on eigenvector centrality.

We additionally consider abstractive baselines to illustrate how well these systems perform on multi-document summarization: *Opinosis* [53] focuses on creating a word co-occurrence graph from the source documents and searching for salient graph paths to create an abstract; *Extract+Rewrite* [168] selects sentences using LexRank and condenses each sentence to a title-like summary; *Pointer-Gen* [152] seeks to generate abstracts by copying words from the source documents and generating novel words not present in the source text.

Our DPP-based framework belongs to a strand of optimization-based methods. In particular, *ICSISumm* (Gillick et al., 2009) formulates extractive summarization as integer linear program-

Table 6.3: Example system summaries and the human reference summary. LexRank extracts long and comprehensive sentences that yield high graph centrality. Pointer-Gen (abstractive) has difficulty in generating faithful summaries (see the last bullet "*all 3-year-olds ... have been given to a child*"). DPP is able to select a balanced set of representative and diverse sentences.

**LexRank Summary**

- The official, Dr. Charles J. Ganley, director of the office of nonprescription drug products at the Food and Drug Administration, said in an interview that the agency was "revisiting the risks and benefits of the use of these drugs in children" and that "we're particularly concerned about the use of these drugs in children less than 2 years of age."
- The Consumer Healthcare Products Association, an industry trade group that has consistently defended the safety of pediatric cough and cold medicines, recommended in its own 156-page safety review, also released Friday, that the FDA consider mandatory warning labels saying that they should not be used in children younger than two.
- Major makers of over-the-counter infant cough and cold medicines announced Thursday that they were voluntarily withdrawing their products from the market for fear that they could be misused by parents.

**Pointer-Gen Summary**

- Dr. Charles Ganley, a top food and drug administration official, said the agency was "revisiting the risks and benefits of the use of these drugs in children," the director of the FDA's office of nonprescription drug products.
- The FDA will formally consider revising labeling at a meeting scheduled for Oct. 18-19.
- The withdrawal comes two weeks after reviewing reports of side effects over the last four decades, a 1994 study found that more than a third of all 3-year-olds in the United States were estimated to have been given to a child.

**DPP-Combined Summary**

- Johnson & Johnson on Thursday voluntarily recalled certain infant cough and cold products, citing "rare" instances of misuse leading to overdoses.
- Federal drug regulators have started a broad review of the safety of popular cough and cold remedies meant for children, a top official said Thursday.
- Safety experts for the Food and Drug Administration urged the agency on Friday to consider an outright ban on over-the-counter, multi-symptom cough and cold medicines for children under 6.
- Major makers of over-the-counter infant cough and cold medicines announced Thursday that they were voluntarily withdrawing their products from the market for fear that they could be misused by parents.

**Human Reference Summary**

- On March 1, 2007, the Food/Drug Administration (FDA) started a broad safety review of children's cough/cold remedies.
- They are particularly concerned about use of these drugs by infants.
- By September 28th, the 356-page FDA review urged an outright ban on all such medicines for children under six.
- Dr. Charles Ganley, a top FDA official said "We have no data on these agents of what's a safe and effective dose in Children." The review also stated that between 1969 and 2006, 123 children died from taking decongestants and antihistimines.
- On October 11th, all such infant products were pulled from the markets.

Table 6.4: Sentence similarity datasets and CapsNet's performance on them. SNLI discriminates between entailment and contradiction; STS is pretrained using Src-Summ pairs and fine-tuned on its train split.

| **Dataset** | **Train** | **Dev** | **Test** | **Accu.** |
|---|---|---|---|---|
| STS-Benchmark [15] | 5,749 | 1,500 | 1,379 | 64.7% |
| SNLI [9] | 366,603 | 6,607 | 6,605 | 93.0% |
| Src-Summ Pairs (this work) | 2,084,798 | 105,936 | 86,144 | **94.8**% |

ming; it identifies a globally-optimal set of sentences covering the most important concepts of the source documents; *DPP* [100] selects an optimal set of sentences that are representative of the source documents and with maximum diversity, as determined by the determinantal point process. Gong et al. [62] show that the DPP performs well on summarizing both text and video.

We experiment with several variants of the DPP model: *DPP-Capsnet* computes the similarity between sentences ($S_{ij}$) using the CapsNet described in Sec. §6.2 and trained using our newly-constructed sentence redundancy dataset, whereas the default DPP framework computes sentence similarity as the cosine similarity of sentence TF-IDF vectors. *DPP-Combined* linearly combines the cosine similarity with the CapsNet output using an interpolation coefficient determined on the dev set[3].

Table 6.1 and 6.2 illustrate the summarization results we have obtained for the DUC-04 and TAC-11 datasets. Our DPP methods perform superior to both extractive and abstractive baselines,

[3]The Capsnet coefficient $\lambda_c$ is selected to be 0.2 and 0.1 respectively for the DUC-04 and TAC-11 dataset.

Table 6.5: Example positive (✓) and negative (✗) sentence pairs from the semantic similarity datasets.

| |
|---|
| **STS-Benchmark (a)** *Four girls happily walk down a sidewalk.* **(b)** *Three young girls walk down a sidewalk.* ✗ |
| **SNLI (a)** *3 young man in hoods standing in the middle of a quiet street facing the camera.* **(b)** *Three hood wearing people pose for a picture.* ✓ |
| **Src-Summ Pairs (a)** *He ended up killing five girls and wounding five others before killing himself.* **(b)** *Nearly four months ago, a milk delivery-truck driver lined up 10 girls in a one-room schoolhouse in this Amish farming community and opened fire, killing five of them and wounding five others before turning the gun on himself.* ✓ |

indicating the effectiveness of optimization-based methods for extractive multi-document summarization. The DPP optimizes for summary sentence selection to maximize their content coverage and diversity, expressed as the squared volume of the space spanned by the selected sentences.

Further, we observe that the DPP system with combined similarity metrics yields the highest performance, achieving 10.14% and 10.13% F-scores respectively on DUC-04 and TAC-11. This finding suggests that the cosine similarity of sentence TF-IDF vectors and the CapsNet semantic similarity successfully complement each other to provide the best overall estimate of sentence redundancy. A close examination of the system outputs reveal that important topical words (e.g.,

"*$3 million*") that are frequently discussed in the document cluster can be crucial for determining sentence redundancy, because sentences sharing the same topical words are more likely to be considered redundant. While neural models such as the CapsNet rarely explicitly model word frequencies, the TF-IDF sentence representation is highly effective in capturing topical terms.

In Table **??** we show example system summaries and a human-written reference summary. We observe that LexRank tends to extract long and comprehensive sentences that yield high graph centrality; the abstractive pointer-generator networks, despite the promising results, can sometimes fail to generate meaningful summaries (e.g., "*a third of all 3-year-olds $\cdots$ have been given to a child*"). In contrast, our DPP method is able to select a balanced set of representative and diverse summary sentences. We next compare several semantic similarity datasets to gain a better understanding of modeling sentence redundancy for summarization.

### 6.4.2 Sentence Similarity

We compare three standard datasets used for semantic similarity tasks, including *SNLI* [9], used for natural language inference, *STS-Benchmark* [15] for semantic equivalence, and our newly-constructed *Src-Summ* sentence pairs. Details are presented in Table 6.4.

We observe that CapsNet achieves the highest prediction accuracy of 94.8% on the *Src-Summ* dataset and it yields similar performance on *SNLI*, indicating the effectiveness of CapsNet on characterizing semantic similarity. *STS* appears to be a more challenging task, where CapsNet yields 64.7% accuracy. Note that we perform two-way classification on SNLI to discriminate

entailment and contradiction. The STS dataset is too small to be used to train CapsNet without overfitting, we thus pre-train the model on *Src-Summ* pairs, and use the train split of *STS* to fine-tune parameters.

Table 6.5 shows example positive and negative sentence pairs from the *STS*, *SNLI*, and *Src-Summ* datasets. The *STS* and *SNLI* datasets are constructed by human annotators to test a system's capability of learning sentence representations. The sentences can share very few words in common but still express an entailment relationship (positive); or the sentences can share a lot of words in common yet they are semantically distinct (negative). These cases are usually not seen in summarization datasets containing clusters of documents discussing single topics. The *Src-Summ* dataset successfully strike a balance between sharing common words yet containing diverse expressions. It is thus a good fit for training classifiers to detect sentence redundancy.

Figure 6.3 compares heatmaps generated by computing cosine similarity of sentence TF-IDF vectors (*Cosine*), and training CapsNet on *SNLI* and *Src-Summ* datasets respectively. We find that the *Cosine* similarity scores are relatively strict, as a vast majority of sentence pairs are assigned zero similarity, because these sentences have no word overlap. At the other extreme, *CapsNet+SNLI* labels a large quantity of sentence pairs as false positives, because its training data frequently contain sentences that share few words in common but nonetheless are positive, i.e., expressing an entailment relationship. The similarity scores generated by *CapsNet+SrcSumm* are more moderate comparing to *CapsNet+SNLI* and *Cosine*, suggesting the appropriateness of using *Src-Summ* sentence pairs for estimating sentence redundancy.

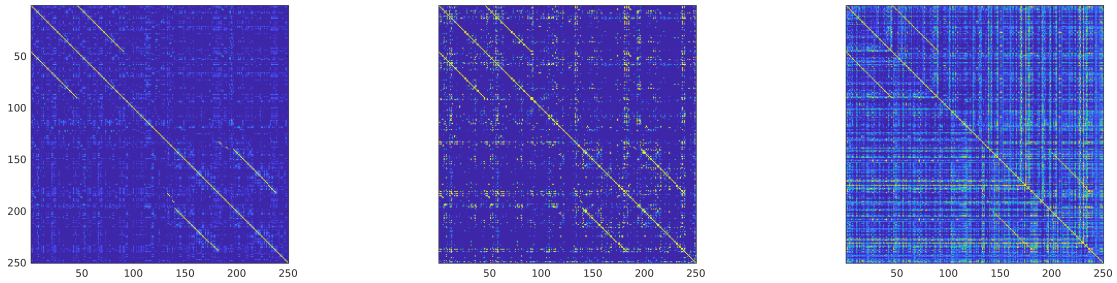



Figure 6.3: Heatmaps for topic D31008 of DUC-04 (cropped to 200 sentences) that shows the cosine similarity score of sentence TF-IDF vectors (*Cosine*, left), and the CapsNet output trained respectively on *SNLI* (right) and *Src-Summ* (middle) datasets. The short off-diagonal lines are near-identical sentences found in the document cluster.

## 6.5 Conclusion

We strengthen a DPP-based multi-document summarization system with improved similarity measure inspired by capsule networks for determining sentence redundancy. We show that redundant sentences not only have common words but they can be semantically similar with little word overlap. Both aspects should be modelled in calculating pairwise sentence similarity. Our system yields competitive results on benchmark datasets surpassing strong summarization baselines.

# CHAPTER 7
# MULTI-DOCUMENT SUMMARIZATION WITH DETERMINANTAL POINT PROCESSES AND CONTEXTUALIZED REPRESENTATIONS

## 7.1 DPP for Summarization

Determinantal point process (Kulesza and Taskar, 2012) defines a probability measure $\mathscr{P}$ over all subsets ($2^{|\mathscr{Y}|}$) of a ground set containing all document sentences $\mathscr{Y} = \{1, 2, \cdots, \mathsf{N}\}$. Our goal is to identify a most probable subset $Y$, corresponding to an extractive summary, that achieves the highest probability score. The probability measure $\mathscr{P}$ is defined as

$$\mathscr{P}(Y; L) = \frac{\det(L_Y)}{\det(L+I)}, \tag{7.1}$$

$$\sum_{Y \subseteq \mathscr{Y}} \det(L_Y) = \det(L+I), \tag{7.2}$$

where $\det(\cdot)$ is the determinant of a matrix; $I$ is the identity matrix; $L \in \mathbb{R}^{\mathsf{N} \times \mathsf{N}}$ is a positive semi-definite (PSD) matrix, known as the $L$-ensemble; $L_{ij}$ indicates the correlation between sentences $i$ and $j$; and $L_Y$ is a submatrix of $L$ containing only entries indexed by elements of $Y$. As illustrated in Eq. (7.1), the probability of an extractive summary $Y \subseteq \mathscr{Y}$ is thus proportional to the determinant of the matrix $L_Y$.

Kulesza and Taskar [101] introduce a decomposition of the $L$-ensemble matrix: $L_{ij} = q_i \cdot S_{ij} \cdot q_j$ where $q_i \in \mathbb{R}^+$ is a positive number indicating the *quality* of a sentence and $S_{ij}$ is a measure

of *similarity* between sentences $i$ and $j$. The $q$ and $S$ model the sentence quality and pairwise similarity respectively and contribute to the $L$-ensemble matrix. A log-linear model is used to determine sentence quality: $q_i = \exp(\theta^\top \mathbf{f}(i))$, where $\mathbf{f}(i)$ is a feature vector for sentence $i$ and $\theta$ are feature weights to be learned during DPP training. We optimize $\theta$ by maximizing log-likelihood with gradient descent, illustrated as follows:

$$\mathscr{L}(\theta) = \sum_{m=1}^{M} \log \mathscr{P}(\hat{Y}^{(m)}; L^{(m)}(\theta)), \tag{7.3}$$

$$\nabla_\theta = \sum_{m=1}^{M} \sum_{i \in \hat{Y}^{(m)}} \mathbf{f}(i) - \sum_{j} \mathbf{f}(j) K_{jj}^{(m)}, \tag{7.4}$$

where $M$ is the total number of training instances; $\hat{Y}^{(m)}$ is the ground-truth summary of the $m$-th instance; $K = L(L+I)^{-1}$ is the kernel matrix and $\mathscr{P}(\hat{Y}^{(m)}; L^{(m)}(\theta))$ is defined by Eq. (7.1). We refer the reader to [101] for details on gradient derivation (Eq. (7.4)). In the following we describe two BERT models to respectively estimate sentence pairwise similarity and importance. The trained models are then plugged into the DPP framework for computing $S$ and $q$.

### 7.1.1 BERT Architecture

We introduce two models that fine-tune the BERT-base architecture [31] to calculate the similarity between a pair of sentences (BERT-*sim*) and learn representations that characterize the importance of a single sentence (BERT-*imp*). Importantly, training instances for both BERT models are derived from *single-document* summarization dataset [67] by Lebanoff et al. [107], containing a collection

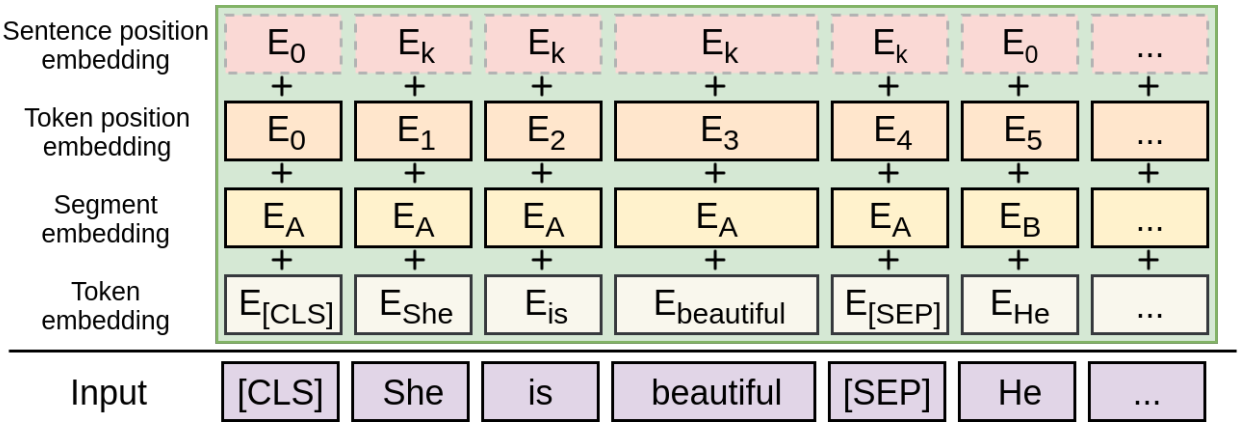



Figure 7.1: Position of summary-worthy sentences in a document for single-doc (CNN/DM) and multi-doc datasets (DUC-04, TAC11). 'pos' are summary-worthy document sentences; 'neg' are sentences that are randomly sampled from the same document.

Table 7.1: BERT-*sim* and BERT-*imp* utilize embeddings for tokens, segments, token position in a sentence and sentence position in a document. These embeddings are element-wisely added up then fed into the model.

| CNN/DM | mean | min | max |
|---|---|---|---|
| train-pos | 13.95 | 1 | 318 |
| train-neg | 21.90 | 1 | 337 |
| DUC-04 | 2.22 | 1 | 5 |
| TAC-11 | 1.67 | 1 | 5 |

of single sentences (or sentence pairs) and their associated labels. During testing, the trained BERT

models are applied to single sentences and sentence pairs derived from *multi-document* input to obtain quality and similarity measures.

BERT-*sim* takes as input a pair of sentences and transforms each token in the sentence into an embedding using an embedding layer. They are then passed through the BERT-base architecture to produce a vector representing the input sentence pair. The vector, denoted by $\mathbf{u} \in \mathbb{R}^d$, is the final hidden state corresponding to the "[CLS]" token ($d$=768), which is used as the aggregate sequence representation. $\mathbf{u}$ is passed through a feed-forward layer with the same dimension $d$, followed by a dropout layer, and a final softmax prediction layer to classify whether a pair of sentences contain redundant information or not. Once the model is trained, we can apply it to a pair of sentences $i$ and $j$ to obtain the similarity score $S_{ij}$.

BERT-*imp* uses a similar architecture to predict if any single sentence is important to the summary. Once the model is trained, we can apply it to the $i$-th sentence to generate a vector $\mathbf{u}_i$ which is used as the feature representation $\mathbf{f}(i)$ for the $i$-th sentence when computing $q_i$.

The embedding layer, illustrated in Fig. 7.1, consists of several types of embeddings, respectively representing tokens, segments, the token position in a sentence and sentence position within a given document. These embeddings are element-wisely added up then fed to the model. The sentence position embeddings are incorporated in this work to capture the position of a sentence in the article. It is utilized only by BERT-*imp*, as position matters for sentence importance but not quite so for pairwise similarity. As shown in Table 7.1, positive sentences in the training data (see §7.2.1) tend to appear at the beginning of an article, consistently more so than negative sentences. Further, ground-truth summary sentences of the DUC and TAC datasets are likely to appear among

the first five sentences of an article, indicating position embeddings are crucial for training the BERT-*imp* model.

### 7.1.2 DPP Training

DPP training focuses on estimating the weights of features used in $q_i = \exp(\theta^\top \mathbf{f}(i))$, which is a log-linear model used for computing sentence quality. The sentence similarity scores $S_{ij}$ are produced by BERT-*sim*; they do not change during DPP training. We obtain contextualized representations for the $i$-th sentence, i.e., $\mathbf{f}(i) \in \mathbb{R}^d$, from the penultimate layer ($\mathbf{u}_i$) of BERT-*imp*.

In addition, a number of surface indicators[1], denoted by $\mathbf{v}_i \in \mathbb{R}^{d'}$, are extracted for sentence $i$. To combine surface indicators and contextualized representations, we concatenate $\mathbf{u}_i$ and $\mathbf{v}_i$ as sentence features. We also take a weighted average[2] of $S_{ij}$ and $C_{ij}$ as an estimate of pairwise sentence similarity, where $C_{ij}$ is the cosine similarity of sentence TF-IDF vectors. DPP training learns feature weights $\theta \in \mathbb{R}^D$, where $D = d + d'$ if the sentence features are concatenated, otherwise $D = d$. DPP is trained on multi-document summarization data with gradient descent (Eq. (7.4)).

[1]The sentence features include the length and position of a sentence, the cosine similarity between sentence and document TF-IDF vectors [100]. We abstain from using sophisticated features to avoid model overfitting.

[2]The coefficient is set to be 0.9 for both datasets.

## 7.2 Experiments

In this section we describe the dataset used to train the BERT-*sim* and BERT-*imp* models, benchmark datasets for multi-document summarization, and experimental settings. Our system shows competitive results comparing to state-of-the-art methods. Example summaries are provided to demonstrate the effectiveness of the proposed method.

### 7.2.1 Dataset

**CNN / DailyMail** This dataset [67] is utilized to train the BERT-*sim* and BERT-*imp* models. For BERT-*sim*, we pair each human summary sentence with its most similar document sentence to create a positive instance; negative instances are randomly sampled sentence pairs. For BERT-*imp*, the most similar document sentence receives a label of 1; randomly sampled sentences are labelled as 0. In total, our training / dev / test sets contain 2,084,798 / 105,936 / 86,144 sentence pairs and the instances are balanced.

**DUC/TAC** We evaluate our DPP approach (§7.1) on multi-document summarization datasets including DUC and TAC [137, 28]. The task is to generate a summary of 100 words from a collection of news articles. We report ROUGE F-scores [114][3] on DUC-04 (trained on DUC-03) and TAC-11 (trained on TAC-08/09/10) following standard settings [72]. Ground-truth extractive summaries used in DPP training are obtained from Cho et al. [23].

[3] with options -n 2 -m -w 1.2 -c 95 -r 1000 -l 100

### 7.2.2 Experiment Settings

We implement our system using TensorFlow on an NVIDIA 1080Ti GPU. We consider the maximum length of a sentence to be 64 or 128 words. The batch size is 64 for the 64 max sentence length and 32 for 128. We use Adam optimizer [92] with the default setting and set learning rate to be 2e-5. We train BERT-*imp* and BERT-*sim* on CNN/DM. The prediction accuracy of BERT-*sim* and BERT-*imp* (with length-128) are respectively 96.11% and 69.05%. Similar results are observed with length-64: 95.79% and 69.63%.

### 7.2.3 Summarization Results

We compare our system with strong summarization baselines (Table 7.2 and 7.3). *SumBasic* [194], *KL-Sum* [64], and *LexRank* [42] are extractive approaches; *Opinosis* [53], *Extract+Rewrite* [168], and *Pointer-Gen* [152] are abstractive methods; *ICSISumm* [59] is an ILP-based summarization method; and *DPP-Caps-Comb*, *DPP-Caps* are results combining DPP and capsule networks reported by Cho et al. [23] w/ and w/o using sentence TF-IDF similarity ($C_{i,j}$).

We experiment with variants of our DPP model: *DPP-BERT*, *DPP-BERT-Combined*. The former utilizes the outputs from BERT-*sim* and BERT-*imp* to compute $S_{ij}$ and $q_i$, whereas the latter combines BERT-*sim* output with sentence TF-IDF similarity ($C_{i,j}$), and concatenates BERT-*imp* features with linguistically informed features.

Table 7.2: Results on the DUC-04 dataset evaluated by ROUGE. † indicates our reimplementation of Kulesza and Taskar [101] system.

| | DUC-04 | | |
|---|---|---|---|
| **System** | **R-1** | **R-2** | **R-SU4** |
| Opinosis [53] | 27.07 | 5.03 | 8.63 |
| Extract+Rewrite [168] | 28.90 | 5.33 | 8.76 |
| Pointer-Gen [152] | 31.43 | 6.03 | 10.01 |
| SumBasic [194] | 29.48 | 4.25 | 8.64 |
| KLSumm(Haghighi et al., 2009) | 31.04 | 6.03 | 10.23 |
| LexRank [42] | 34.44 | 7.11 | 11.19 |
| ICSISumm [58] | 37.31 | 9.36 | 13.12 |
| DPP [101]† | 38.10 | 9.14 | 13.40 |
| DPP-Caps [23] | 38.25 | 9.22 | 13.40 |
| DPP-Caps-Comb [23] | 39.35 | 10.14 | 14.15 |
| DPP-BERT (ours) | 38.14 | 9.30 | 13.47 |
| DPP-BERT-Comb 64 (ours) | 38.78 | 9.78 | 14.04 |
| DPP-BERT-Comb 128 (ours) | **39.05** | **10.23** | **14.35** |

Our DPP methods outperform both extractive and abstractive baselines, indicating the effectiveness of optimization-based methods for extractive multi-document summarization. Furthermore, we observe that *DPP-BERT-Combined* yields the best performance, achieving 10.23% and 11.06% F-scores respectively on DUC-04 and TAC-11. This finding suggests that sentence similarity scores and importance features from the *DPP-BERT* system and TF-IDF based features can

Table 7.3: ROUGE results on the TAC-11 dataset.

| | TAC-11 | | |
|---|---|---|---|
| **System** | **R-1** | **R-2** | **R-SU4** |
| Opinosis [53] | 25.15 | 5.12 | 8.12 |
| Extract+Rewrite [168] | 29.07 | 6.11 | 9.20 |
| Pointer-Gen [152] | 31.44 | 6.40 | 10.20 |
| SumBasic [194] | 31.58 | 6.06 | 10.06 |
| KLSumm (Haghighi et al., 2009) | 31.23 | 7.07 | 10.56 |
| LexRank [42] | 33.10 | 7.50 | 11.13 |
| DPP [101]† | 36.95 | 9.83 | 13.57 |
| DPP-Caps [23] | 36.61 | 9.30 | 13.09 |
| DPP-Caps-Comb [23] | 37.30 | 10.13 | 13.78 |
| DPP-BERT (ours) | 37.04 | 10.18 | 13.79 |
| DPP-BERT-Comb 64 (ours) | 38.46 | 10.79 | 14.45 |
| DPP-BERT-Comb 128 (ours) | **38.59** | **11.06** | **14.65** |

complement each other to boost system performance. We conjecture that TF-IDF sentence vectors are effective at representing topical terms (e.g., *3 million*), thus helping DPP better select representative sentences. Another observation is that *DPP-BERT* and *DPP-BERT-Combined* consistently outperform *DPP-Caps* and *DPP-Caps-Comb*, indicating its excellence for DPP-based summarization.

In Table 7.4 we show example system summaries and a human-written reference summary. *DPP-BERT* and *DPP-BERT-Combined* both are capable of selecting a balanced set of represen-

tative and diverse summary sentence from multi-documents. *DPP-BERT-Combined* selects more relevant sentences than *DPP-BERT* comparing to the human summary, leading to better ROUGE scores.

## 7.3 Conclusion

In this paper we describe a novel approach using determinantal point processes for extractive multi-document summarization. Our DPP+BERT models harness the power of deep contextualized representations and optimization to achieve outstanding performance on multi-document summarization benchmarks. Our analysis further reveals that, despite the success of deep contextualized representations, it remains necessary to combine them with surface indicators for effective identification of summary-worthy sentences.

Table 7.4: Example system summaries and their human reference summary. Sentences selected by DPP-BERT-Combined are more similar to the human summary than those of DPP-BERT; both include diverse sentences.

**Human Reference Summary**

- On March 1, 2007, the Food/Drug Administration (FDA) started a broad safety review of children's cough/cold remedies.
- They are particularly concerned about use of these drugs by infants.
- By September 28th, the 356-page FDA review urged an outright ban on all such medicines for children under six.
- Dr. Charles Ganley, a top FDA official said "We have no data on these agents of what's a safe and effective dose in Children." The review also stated that between 1969 and 2006, 123 children died from taking decongestants and antihistimines.
- On October 11th, all such infant products were pulled from the markets.

**DPP-BERT Summary**

- The petition is far from the first warning about children using the medicines.
- The FDA will formally consider revising labeling at a meeting scheduled for Oct. 18-19.
- Federal drug regulators have started a broad review of the safety of popular cough and cold remedies meant for children, a top official said Thursday.
- Similarly, hydrocodone has never been shown to be safe and effective in children, and its dangers as a powerful and potentially addictive narcotic are clear.

**DPP-BERT-Combined Summary**

- The U.S. government is warning parents not to give cough and cold medicines to children under 2 without a doctor's order, part of an overall review of the products' safety and effectiveness for youngsters.
- Drug makers on Thursday voluntarily pulled kids' cold medicines off the market less than two weeks after the U.S. government warned of potential health risks to infants.
- Safety experts for the Food and Drug Administration urged the agency on Friday to consider an outright ban on over-the-counter, multi-symptom cough and cold medicines for children under 6.
- In high doses, cold medicines can affect the heart's electrical system, leading to arrhythmias.

# CHAPTER 8
# BETTER HIGHLIGHTING: CREATING SUB-SENTENCE SUMMARY HIGHLIGHTS

## 8.1 Method for Creating Sub-Sentence Segments

We present a new method to identify self-contained segments, then select important and non-redundant segments to form a summary, as text fragments containing incomplete and disorganized information are hardly successful summary highlights.

### 8.1.1 Self-Contained Segments

A self-contained segment is, in a sense, a miniature sentence; a sentence containing incomplete or ungrammatical constructions is incomprehensible to human inspection. Table 2.1 shows examples of self-contained and non-self-contained segments. Since its very inception [199], the concept of "semantically self-contained segment" has not been sufficiently examined in the literature and lacks an universal definition. We argue in this paper that a self-contained segment shall conform to certain syntactic validity constraints and there exists only weak dependencies between words that belong to the segment and those do not.

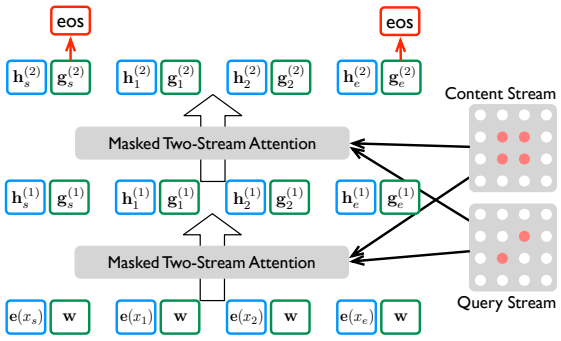



Figure 8.1: The XLNet architecture with two-stream attention mechanism is leveraged to estimate whether a segment is self-contained or not. A self-contained segment is assumed to be preceded and followed by end-of-sentence markers (eos).

The automatic identification of self-contained segments requires more than segmentation or parsing sentences into tree structures [37]. Self-contained segments do not necessarily correspond to constituents in the tree and further, there is no guarantee that tree constituents are self-contained. In this paper, we define a segment to be a consecutive sequence of words, excluding segments formed by concatenating non-adjacent words from consideration. We perform exhaustive search to analyze every segment of a given sentence to determine if it is self-contained or not.

Let $\mathbf{x} = [x_1, \ldots, x_N]$ be a document sentence. We present a method to estimate whether an arbitrary segment $\mathbf{x}_{i:j}$ of the sentence is semantically self-contained or not. Our method is inspired by XLNet [225] that introduces a novel architecture with two-stream attention mechanism for autoregressive language modeling. Pretrained contextualized representations such as BERT and XLNet have demonstrated remarkable success on language understanding tasks. We expect the

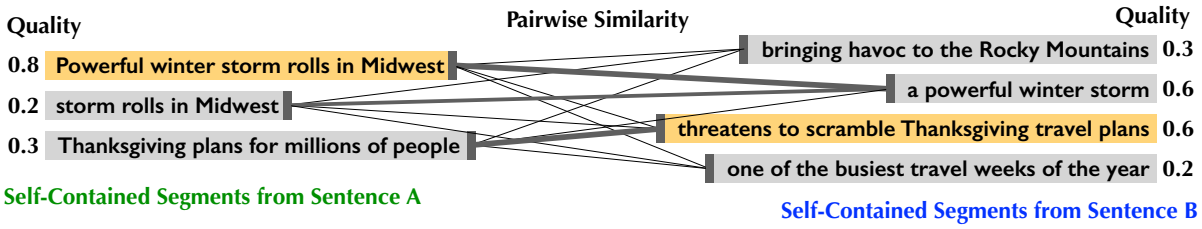



Figure 8.2: DPP selects a set of summary segments (marked yellow) based on the *quality* and *pairwise dissimilarity* of segments.

representations to encode the syntactic validity of segments, as similar findings are seen in recent structural probings [68].

We hypothesize that a self-contained segment, similar to a miniature sentence, can be preceded and followed by end-of-sentence (eos) markers without sacrificing grammatical correctness. We follow the convention of Clark et al. [25] of defining end-of-sentence markers (eos) to include periods and commas. Our method inserts hypothetical tokens $x_{\text{s}}$ and $x_{\text{e}}$ to the beginning and end positions of a segment $\mathbf{x}_{i:j}$, then constructs contextualized representations for these positions, denoted by $\mathbf{g}(\mathbf{x}_{i:j}, p_{\text{start}})$ and $\mathbf{g}(\mathbf{x}_{i:j}, p_{\text{end}})$, based on which we estimate how likely $x_{\text{s}}$ is an end-of-sentence marker $p(x_{\text{s}}{=}\text{eos}|\mathbf{x}_{i:j})$, similarly for $p(x_{\text{e}}{=}\text{eos}|\mathbf{x}_{i:j})$. Their average probability indicates self-containedness. A higher score of $p(z|\mathbf{x}_{i:j})$ suggests $\mathbf{x}_{i:j}$ has a higher likelihood of being self-contained.

$$p(z|\mathbf{x}_{i:j}) = \frac{1}{2}\Big(p(x_{\text{s}}{=}\text{eos}|\mathbf{x}_{i:j}) + p(x_{\text{e}}{=}\text{eos}|\mathbf{x}_{i:j})\Big)$$

$$p(x_{\text{s}}{=}\text{eos}|\mathbf{x}_{i:j}) = \frac{\exp(\mathbf{e}(x_{\text{s}})^{\top}\mathbf{g}(\mathbf{x}_{i:j}, p_{\text{start}}))}{\sum_{x'}\exp(\mathbf{e}(x')^{\top}\mathbf{g}(\mathbf{x}_{i:j}, p_{\text{start}}))}$$

It is important to induce contextualized representations for the augmented segment *without* using the content of hypothetical tokens $x_{\text{s}}$ and $x_{\text{e}}$. We leverage XLNet with two-stream attention mechanism for this purpose, as illustrated in Figure 8.1. For the $k$-th position ($k$={i:j, start, end}) of the $l$-th layer, a *content* stream builds representation $\mathbf{h}_k^{(l)}$ by attending to all tokens of the segment, whereas a *query* stream builds representation $\mathbf{g}_k^{(l-1)}$ simultaneously without incorporating the content of the current token $x_k$, following the equations given below. Our method builds on the pretrained XLNet model without fine-tuning. It relies on two-stream attention to construct deep contextualized representations $\mathbf{g}(\mathbf{x}_{i:j}, p_{\text{start}})$ and $\mathbf{g}(\mathbf{x}_{i:j}, p_{\text{end}})$, respectively for the beginning and end positions.

$$\mathbf{h}_k^{(l)} = \text{Attention}(\text{Q} = \mathbf{h}_k^{(l-1)}, \text{KV} = \mathbf{h}_{i:j}^{(l-1)})$$

$$\mathbf{g}_k^{(l)} = \text{Attention}(\text{Q} = \mathbf{g}_k^{(l-1)}, \text{KV} = \mathbf{h}_{i:j\backslash k}^{(l-1)})$$

Our method is the first attempt to extract semantically *self-contained* segments from whole sentences. Segments that do not resemble "miniature sentences" will be given low probabilities by the method. E.g., "*closed and hundreds of flights have been*" is scored low, not only because an end-of-sentence marker rarely occurs after "have been," but also the syntactic structure of the segment does not resemble that of a well-formed sentence.

We split a sentence at punctuation and extract a number of segments from each sentence chunk. A segment is discarded if its start (or end) probability is lower than the upper quartile value, indicating an inappropriate start (or end) point. The remaining segments are ordered according to the

average probability. This process produces a collection of self-contained and partially-overlapping segments from a set of documents. Next, we assess the informativeness of the segments and leverage DPP to identify a subset to form the summary highlights.

### 8.1.2 Segment Selection with DPP

We employ the modeling framework proposed by Cho et al. [24] to model determinantal point processes. DPP [101] defines a probability measure $\mathscr{P}$ over all subsets ($2^{|\mathscr{Y}|}$) of a ground set containing a collection of $\mathsf{N}$ segments $\mathscr{Y} = \{1, 2, \cdots, \mathsf{N}\}$. The probability of an extractive summary, containing a subset of the segments $Y \subseteq \mathscr{Y}$, is defined by Eq. (8.1), where $\det(\cdot)$ is the determinant of a matrix; $L \in \mathbb{R}^{\mathsf{N}\times\mathsf{N}}$ is a positive semi-definite matrix and $L_{ij}$ indicates the correlation between segments $i$ and $j$; $L_Y$ is a submatrix of $L$ containing only entries indexed by elements in $Y$; $I$ is the identity matrix. This definition suggests that the probability of a summary $\mathscr{P}(Y;L)$ is proportional to the determinant of $L_Y$.

$$\mathscr{P}(Y;L) = \frac{\det(L_Y)}{\det(L+I)}, \tag{8.1}$$

$$\mathscr{L}(\theta) = \sum_{i=1}^{N} \log \mathscr{P}(\hat{Y}^{(i)}; L^{(i)}(\theta)) \tag{8.2}$$

A decomposition exists for the $L$-ensemble matrix: $L_{ij} = q_i \cdot S_{ij} \cdot q_j$ where $q_i \in \mathbb{R}^+$ is a quality score of the $i$-th segment and $S_{ij}$ is a pairwise similarity score between segments $i$ and $j$. If $q$ and $S$ are available, $\mathscr{P}(Y)$ can be computed using Eq. (8.1). Estimating the pairwise similarity $S$ is trivial, we refer the reader to [24] for details. In this paper, we present a *inverted pyramid* method to

estimate the quality of segments $q$. The quality model is parameterized by $\theta$, thus the L-ensemble is parameterized the same, denoted by $L^{(i)}(\theta)$ for the $i$-th instance of the dataset. $\hat{Y}^{(i)}$ represents the ground-truth summary (Eq. (8.2)). The model is optimized by maximizing the log-likelihood, where parameters $\theta$ are learned during training. As illustrated in Figure 8.2, DPP allows us to identify a set of salient and non-redundant summary segments.

**Inverted pyramid** We describe a classifier to predict if a segment of text is summary-worthy or not according to the *inverted pyramid* principle.[1] It is a way of front loading a story so that the reader can get the most important information first. E.g., the most newsworthy information such as who, what, when, where, etc. heads the article, followed by important details, and finally other general and background information. The inverted pyramid explains the common observation that lead baselines consisting of the first few sentences of an article perform strongly in the news domain.

Our classifier assigns a high score to a segment if its content is relevant to the lead paragraph, and a low score if its content overlaps with the bottom paragraph of a news article, which usually contains trivial details. Importantly, the classifier is trained using CNN/DM [153], rather than any multi-document summarization data.

During training, we obtain the ground-truth summary of each article. A summary sentence is paired with the lead paragraph of the article that contains the top-5 sentences to form a *positive* instance and similarly, with bottom-5 sentences to form a *negative* instance. If a summary sentence appears as-is in the top or bottom paragraph, we exclude the sentence from the paragraph to avoid

[1] https://en.wikipedia.org/wiki/Inverted_pyramid_(journalism)

overfitting the classifier. At test time, the classifier learns to distill the essential content of the segment and assigns a high score to it if its content is similar to the lead paragraph, indicating the segment is relevant and summary-worthy.

For each instance, we obtain deep contextualized representation for it using the BERT architecture, where a segment and a lead (or bottom) paragraph is used as the input and the top layer hidden vector of the [CLS] token is extracted as the representation. It is fed to a feedforward, a dropout and a softmax layer to predict a binary label for the segment. Once the model is trained, we apply it to a segment and its lead paragraph to produce a vector which is used as part of the features for computing $q$.

**DPP training.** We obtain feature representations for the $i$-th segment by concatenating the previous vector and a number of surface features extracted for segment $i$. The features include the length and position of the segment within a document, the cosine similarity between the segment and document TF-IDF vectors [100]. We abstain from using sophisticated features to avoid model overfitting. The features parameters $\theta$ are to be learned during DPP training.

DPP is trained on multi-document summarization data by maximizing log-likelihood. At each iteration, we project the L-ensemble onto the positive semi-definite (PSD) cone to ensure that it satisfies the PSD property (§8.1.2). This is accomplished in two steps, where $L'$ is the new L-ensemble.

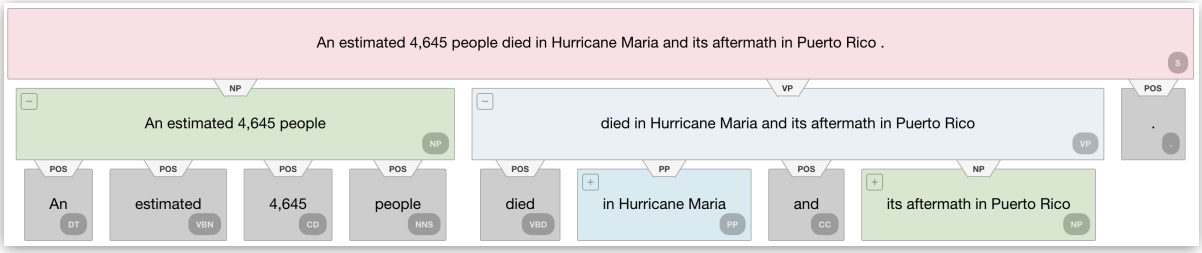



Figure 8.3: Example of a constituent parse tree, from which tree segments are extracted.

$$L = \sum_{i=0}^{n} \lambda_i v_i v_i^\top \text{ (Eigenvalue decomposition)}$$

$$L' = \sum_{i=0}^{n} \max\{\lambda_i, 0\} v_i v_i^\top \text{ (PSD projection)}$$

## 8.2 Experiments

### 8.2.1 Data Sets

Our data comes from NIST. We use them to investigate the feasibility of the proposed multi-document summarization method. Particularly, we use DUC-03/04 [137] and TAC-08/09/10/11 datasets [28], which contain 60/50/48/44/46/44 document sets respectively. These datasets are previously used as benchmarks for multi-document summarization competitions.[2] Our task is to generate a summary of less than 100 words from a set of 10 news documents, where a summary

[2]https://tac.nist.gov/data/
https://duc.nist.gov/data/

contains a set of selected text segments. There are four human reference summaries for each document set, created by NIST evaluators.

A system summary is evaluated against human reference summaries using ROUGE [114][3], where R-1, R-2, and R-SU4 respectively measure the overlap of unigrams, bigrams and skip bigrams (with a maximum gap of 4 words) between system and reference summaries. In the following sections, we report results on DUC-04 (trained on DUC-03) and TAC-11 (trained on TAC-08/09/10) as they are the standard test sets [72].

### 8.2.2 Experimental Settings

Our method for predicting self-containedness uses the pretrained XLNet-LARGE [225] to estimate the probability of end-of-sentence markers. We require a candidate segment to contain five or more words. Our classifier is based on the BERT-BASE model and it is fine-tuned for two epochs on the training data. The maximum sequence length of the model is 512 tokens and the batch size is set to 16. We use the Adam optimizer with an initial learning rate of $5e^{-5}$, a warm-up period of 24,400 steps, corresponding to 10% of the training data, and linear decay after that.

[3] w/ options -n 2 -m -w 1.2 -c 95 -r 1000 -l 100

Table 8.1: Results on DUC-04 dataset evaluated by ROUGE.

| **DUC-04 Test Set** | **R-1** | **R-2** | **R-SU4** |
|---|---|---|---|
| DPP-BERT [24] | 39.05 | 10.23 | 14.35 |
| DPP [101] | 38.10 | 9.14 | 13.40 |
| SumBasic [194] | 29.48 | 4.25 | 8.64 |
| KLSumm(Haghighi et al., 2009) | 31.04 | 6.03 | 10.23 |
| LexRank [42] | 34.44 | 7.11 | 11.19 |
| Centroid [72] | 35.49 | 7.80 | 12.02 |
| ICSISumm [58] | 37.31 | 9.36 | 13.12 |
| Opinosis [53] | 27.07 | 5.03 | 8.63 |
| Pointer-Gen [153] | 31.43 | 6.03 | 10.01 |
| CopyTrans [55] | 28.54 | 6.38 | 7.22 |
| Hi-MAP [43] | 35.78 | 8.90 | 11.43 |
| HL-TreeSegs (Our work) | 39.18 | 10.30 | 14.37 |
| HL-XLNetSegs (Our work) | **39.26** | **10.70** | **14.47** |

Table 8.2: ROUGE results on the TAC-11 dataset.

| **TAC-11 Test Set** | **R-1** | **R-2** | **R-SU4** |
|---|---|---|---|
| DPP-BERT [24] | 38.59 | 11.06 | 14.65 |
| DPP [101] | 36.95 | 9.83 | 13.57 |
| SumBasic [194] | 31.58 | 6.06 | 10.06 |
| KLSumm (Haghighi et al., 2009) | 31.23 | 7.07 | 10.56 |
| LexRank [42] | 33.10 | 7.50 | 11.13 |
| Opinosis [53] | 25.15 | 5.12 | 8.12 |
| Pointer-Gen [153] | 31.44 | 6.40 | 10.20 |
| HL-XLNetSegs (Our work) | 36.50 | 9.76 | 13.34 |
| HL-TreeSegs (Our work) | **37.24** | **10.04** | **13.49** |

### 8.2.3 Ground-Truth Segments

Our DPP framework is fully supervised and ground-truth summary segments are required for training the DPP. In an ideal scenario, we would have human annotators to label the ground-truth summary segments for each document set. It is akin to label bounding boxes for objects, which allows an object detector to be trained on millions of training examples [61]. Nonetheless, human anno-

tation is tedious, expensive and time-consuming. We cannot afford to have human annotators to label a large number of segments.

We introduce an approximation method instead. First, we greedily select a set of summary sentences from a document set that achieve the highest R-2 F-score with human reference summaries. Secondly, for every summary sentence, we identify a single segment from a collection of over-generated and self-contained segments (§8.1.1), such that the selected attains the highest R-2 F-score with human summaries. Such segments are labelled as positive. This two-step process allows for easy generation of ground-truth summary segments.

### 8.2.4 Summarization Results

Highlighting sub-sentence segments is particularly suited for multi-document summarization, as it allows summaries to be understood in context. We compare our method with strong baselines using extractive and abstractive methods, results are shown in Table 8.1 and 8.2. *DPP* [101] and its variant *DPP-BERT* [24] use determinantal point processes to extract *whole sentences* from document sets. *SumBasic* is an extractive approach leveraging the fact that frequently occurring words are more likely to be included in the summary [194]. *KL-Sum* is a greedy approach that iteratively adds sentences to the summary to minimize KL divergence [64]. *LexRank* [42] is a graph-based approach estimating sentence importance based on eigenvector centrality. All of these methods extract whole sentences rather than segments from sets of documents.

Table 8.3: Examples of system output for a topic of DUC-04. Our highlighting method is superior to sentence extraction as it can help readers quickly sift through a large amount of texts to grasp the main points. The XLNet segments are better than subtrees. Not only can they aid reader comprehension but they are also self-contained and more concise.

**Human Abstract**

• Exxon and Mobil discuss combining business operations.

• A possible Exxon-Mobil merger would reunite 2 parts of Standard Oil broken up by the Supreme Court in 1911.

• Low crude oil prices and the high cost of exploration are motives for a merger that would create the world's largest oil company.

• As Exxon-Mobil merger talks continue, stocks of both companies surge.

• The merger talks show that corporate mergers are back in vogue.

• Antitrust lawyers, industry analysts, and government officials say a merger would require divestitures.

• A Mobil employee worries that a merger would put thousands out of work, but notes that his company's stock would go up.

**Highlighting (Tree Segments)**

• Whether or not the talks between Exxon and Mobil lead to a merger or some other business combination, America's economic history is already being rewritten.

• The boards of Exxon Corp. and Mobil Corp. are expected to meet Tuesday to consider a possible merger agreement that would form the world's largest oil company, a source close to the negotiations said Friday.

• Exxon Corp. and Mobil Corp. have held discussions about combining their business operations, a person involved in the talks said Wednesday.

• News that Exxon and Mobil, two giants in the energy patch, were in merger talks last week is the biggest sign yet that corporate marriages are back in vogue. (Rest omitted.)

**Highlighting (XLNet Segments)**

• Whether or not the talks between Exxon and Mobil lead to a merger or some other business combination, America's economic history is already being rewritten.

• Still, it boggles the mind to accept the notion that hardship is driving profitable Big Oil to either merge, as British Petroleum and Amoco have already agreed to do, or at least to consider the prospect, as Exxon and Mobil are doing.

• Oil stocks led the way as investors soaked up the news of continuing talks between Exxon and Mobil on a merger that would create the world's largest oil company.

• Although the companies only confirmed that they were discussing the possibility of a merger, a person close to the discussions said the boards of both Exxon and Mobil were expected to meet Tuesday to consider an agreement.

• Analysts predicted that there would be huge cuts in duplicate staff from both companies, which employ 122,700 people. (Rest omitted.)

We further consider abstractive summarization methods. *Opinosis* [53] creates a word co-occurrence graph and searches for a graph path to generate an abstract. *PointerGen* [153] learns to reuse source words or predict new words. The documents are concatenated to serve as input. *CopyTrans* uses a 4-layer Transformer for the encoder and decoder [55]. *Hi-MAP* introduces an end-to-end hierarchical attention model [43] to generate abstracts from multi-document inputs.

We explore two variants of our proposed method, called *HL-XLNetSegs* and *HL-TreeSegs*, focusing on highlighting summary segments. The former utilizes XLNet to extract a set of partially-overlapping segments from a sentence; the latter decomposes a sentence constituent parse tree into subtrees and collect segments governed by the subtrees. An illustration is shown in Figure 8.3. Constituent parse trees are obtained using the Stanford parser [124]. In both cases, the segments are passed to DPP, which identifies a set of important and non-redundant segments as highlights.

As shown in Tables 8.1 and 8.2, we find both methods to perform competitively when compared to the leading extractive and abstractive systems, while generating segments with simpler structure. Our *HL-XLNetSegs* method achieves the highest scores among all systems on DUC-04 and it achieves comparable results to others on TAC-11. Breaking a sentence into smaller segments expands the search space dramatically, making it a challenging task to accurately identify summary segments. The degree of difficulty involved in generating sub-sentence highlights is thus beyond that of sentence selection. A similar finding is noted in other studies [20].

Table 8.5 presents a direct comparison of XLNet and tree segments on DUC and TAC datasets. We find that XLNet segments are more concise than tree segments. A tree segment contains 13 tokens on average, while an XLNet segment contains 9.6 tokens on DUC-04. Both methods produce

Table 8.4: Examples of segments generated by XLNet and their scores of self-containedness.

| **Segments and Scores of Self-Containedness** | | |
|---|---|---|
| 1. | 0.646 | winter storms hit during one of the year's busiest travel weeks |
| 2. | 0.644 | storms hit during one of the year's busiest travel weeks |
| 3. | 0.584 | of the year's busiest travel weeks |
| 4. | 0.525 | one of the year's busiest travel weeks |
| ... | ... | ... |
| 10. | 0.132 | and hundreds of flights have been canceled as winter storms hit during one of the year's busiest travel weeks |
| 11. | 0.122 | and hundreds of flights have been canceled as winter storms hit |
| ... | ... | ... |
| 150. | 0.0019 | of flights have been canceled as winter |
| 151. | 0.0014 | Some interstates are closed and hundreds of flights have been canceled as winter |
| 152. | 0.0013 | hundreds of flights have been canceled as winter |
| 153. | 0.0008 | are closed and hundreds of flights have been canceled as winter |

a large number of candidate segments, ranging from 350 to 550 segments per document set, with only 9 to 17 ground-truth summary segments per document set. The small ratio poses a substantial challenge for DPP. Not only must it identify salient content but it has to accurately identify the segments worthy of being included in the summary. In Table 8.3, we show example highlighting of both methods; more examples are available in the supplementary.

Table 8.5: Statistics of text segments generated by XLNet and the constituent parse tree method on DUC/TAC datasets.

| | **DUC** | **TAC** |
|---|---|---|
| # Words per XLNet segment | 9.55 | 8.05 |
| # XLNet segments per sentence | 2.48 | 2.49 |
| # Total segments per document set | 398 | 352 |
| # Summary segments per document set | 9.62 | 9.09 |
| # Words per tree segment | 12.89 | 13.94 |
| # Tree segments per sentence | 3.31 | 3.33 |
| # Total segments per document set | 549 | 478 |
| # Summary segments per document set | 13.68 | 16.56 |

Segments generated by XLNet are sorted according to their scores of self-containedness, $p(z|\mathbf{x}_{i:j})$. In Table 8.4, we provide examples of segments and their scores. The higher the score, the more likely the segment resembles a "miniature sentence." We are particularly interested in understanding where the original sentence is placed according to XLNet; this is shown in Figure 8.4. We observe that in 60% of the cases, the original sentence is placed among the top-10 candidates, suggesting the effectiveness of the XLNet model. As segments are shorter and occur more often in natural language texts, it is possible that they are considered more self-contained than the original sentence.

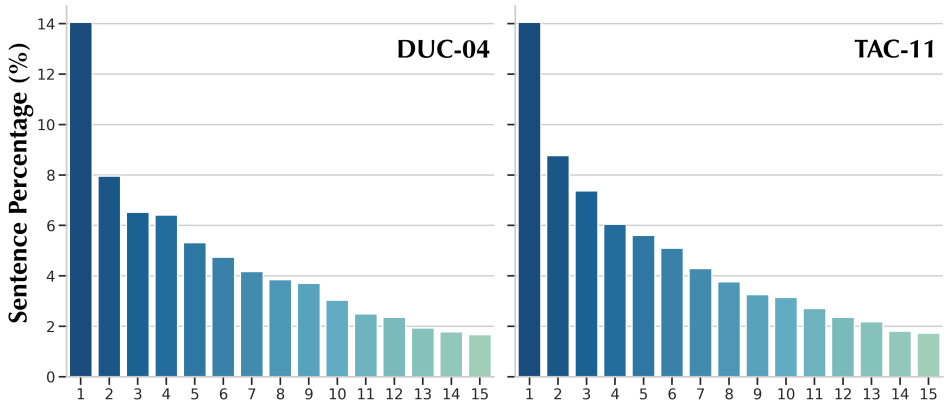



Figure 8.4: Absolute position of the whole sentence among all segments sorted by XLNet scores of self-containedness.

Segments extracted from subtrees are sorted by the depth of tree nodes. The higher nodes are informative constituents denoting complex noun phrases and sentential clauses [76]. An important caveat of the tree segments is their lack of *coverage*. E.g., "*4,645 people died*" is a valid self-contained segment, but it does not belong to a tree constituent, as seen in Figure 8.3. Given that drawback, we focus on segments created by XLNet in our experiments.

### 8.2.5 Self-Containedness

We perform further analysis to investigate the effectiveness of our method on generating self-contained segments (§8.1.1). It is impractical to create a gold-standard by exhaustively enumerating all segments then asking human raters to judge each of them, as the number of segments is polynomial. Instead, we perform post-hoc evaluation on segments generated by our XLNet algo-

Table 8.6: Human evaluation of the self-containedness of text segments. The top-3 segments of XLNet exhibit a high degree of self-containedness: 61% of them have an average score of 3 or above, 34% have ≥4 score, and 12% receive the full score.

| | **Self-Containedness Score** | | | |
|---|---|---|---|---|
| XLNet | ≥3(%) | ≥4(%) | =5(%) | Average |
| All Segments | 54.86 | 30.00 | 10.68 | 2.80 |
| Top-5 Segments | 55.25 | 30.24 | 10.78 | 2.81 |
| Top-3 Segments | **61.04** | **34.04** | **12.42** | **2.95** |

rithm, which are used as input to DPP. We sample 20 topics from TAC-11, with 3 sentences per document for a total of 585 sentences and 1,792 system-generated segments. A human rater is provided with the original sentence and its segments and asked to score each segment on a Likert scale of 1 (worst) to 5 (best) for self-containedness. We employ 5 human raters to judge each segment, the average scores are reported in Table 8.6. We observe that 61% of top-3 segments have an average score of ≥3; 34% have a score ≥4; and 12% receive the full score. The human raters are able to achieve a moderate level of agreement, 44% of the segments have their majority score agreed by three or more raters. Table 8.7 presents example segments and their human assessment scores (more in supplementary). Our sub-sentence segments allow the reader to grasp the main points while remaining succinct and accessible. It thus offers a promising avenue of future research.

Table 8.7: Examples of text segments produced by the XLNet algorithm. Human assessment scores of *self-containedness* are shown in the parentheses (1 being worst & 5 being best).

| |
|---|
| **[Original Sentence] District Attorney David Roger agreed to drop charges including kidnapping, armed robbery, assault with a deadly weapon and conspiracy against both men.** |
| • District Attorney David Roger agreed to drop charges including kidnapping, armed robbery, assault with a deadly weapon and conspiracy against both men. (4.0) |
| • District Attorney David Roger agreed to drop charges including kidnapping, armed robbery, assault with a deadly weapon and conspiracy against both men. (3.8) |
| • District Attorney David Roger agreed to drop charges including kidnapping, armed robbery, assault with a deadly weapon and conspiracy against both men. (3.6) |

## 8.3 Conclusion

We make a first attempt to create sub-sentence summary highlights that are understandable and require minimum information from the surrounding context. Highlighting is important to help readers sift through a large amount of texts and quickly grasp the main points. We describe a novel methodology to generate a rich set of self-contained segments from the documents, then use determinantal point processes to identify summary highlights. The method can be extended to other text genres such as public policies to aid reader comprehension, which will be our future work to explore.

# CHAPTER 9
# CONCLUSION

In this dissertation, we study the problem of sequential data understanding. We hypothesized and demonstrated that indeed sequential data share some common characteristics that would allow one to borrow ideas from one domain to solve problems in others. A great example of this, as we thoroughly discussed throughout this dissertation, is the shared characteristics of video and text. For instance, we modeled videos of human actions as a sequence of "words" that can be explored to discover latent information, in a manner very much similar to text, e.g. in terms of building codebooks, sequence learning, summarization, etc. Thus, this dissertation used two different problems of action recognition and text summarization, in different domains of Computer Vision and Natural Language Processing, to demonstrate and exploit this latent common thread.

In essence, once encoded in the feature space, for understanding and summarizing sequential data, we strive to discover their contextual information, regardless of the modality. The following is a summary of our findings and proposed models on the two sets problems that we studied in this context.

## 9.1 Summary

For action recognition, we present a novel method for learning video-level features in a data-driven manner and evaluate it on large-scale action recognition datasets. The temporal CNNs with different sizes of kernels can extract quality features and are shown to outperform existing approaches. The extracted features contain information on how the input sequence is changed over time, which is the key to attaining the state-of-the-art results.

The two stream network that is employed in many works lacks the spatio-temporal cues for the action recognition task. We propose a fusion network that takes temporal changes of two modalities, appearance and motion, to obtain spatio-temporal features. The proposed network utilizes the temporal CNNs with a residual connection and is applied to low-level features from appearance and motion data to extract temporal information. The network then fuses the two different temporal information to obtain spatio-temporal features. This fusion strategy is shown to be effective for action recognition in two action recognition benchmarks.

The shortcomings of local attention by using CNNs or recurrent neural networks can be overcome with self-attention networks. The self-attention network can correlate short and long term temporal sequences so that a variety of features can be retrieved to understand sequential data. The skeleton-based videos are processed with conventional CNNs to extract low-level features. Then, the self-attention networks extract temporal associations between pairwise features for the action recognition task. A diversity of models utilizing the self-attention networks are introduced that outperform the state-of-the-art approaches in two large-scale datasets.

While the video is a sequence of frames and it is important to extract temporal information, text is a sequence of words and context from each sentence is the key information to understand better the relations among sentences. Our proposed method of extracting context information are combined with the DPP framework to outperform the state of the art in multi-document summarization. For obtaining context, we employ two different methods. The first one is the capsule network that detects transformations of features, so that correlations among word features can be discovered. The association of each word feature is the context in a sentence. The other approach is to employ the language models that are trained on huge amount of text data with the transformer based models. As the data-driven language model performs well on many NLP tasks, and holds rich context information, we use the pretrained network to obtain contextualized representations of each sentence. The proposed models outperform the state of the art on two multi-document benchmarks and show the quality summaries that are faithful representatives, while avoiding redundant sentences.

We make a first attempt in the literature to create sub-sentence summary highlights that are understandable and require minimum information from the surrounding context. Highlighting is important to help readers sift through a large amount of texts and quickly grasp the main points. We describe a novel methodology to generate a rich set of self-contained segments from the documents, then use determinantal point processes to identify summary highlights.

## 9.2 Future Work

In this dissertation, we have shown that sequential data in different modalities can be processed in a similar way. The most important information to be retrieved from sequential data is the latent temporal relation information, or context information. Different methods are introduced to extract the context information: 1D CNN, Self-Attention Network, Capsule Network, and pretrained Language Models. Nevertheless, there is still need for further investigation of different approaches to discover the context information, and potentially many other interesting questions. Follow-up future work to this dissertation may include the following:

The first promising direction is to explore self-supervised or unsupervised methods to learn spatio-temporal context information [184] for the action recognition task. Data driven machine learning methods require more and more data for training, but the annotation cost is expensive and it is hard to get good quality annotations. Thus, unsupervised learning methods without annotation data will contribute to learn more basic principle of human body movement. The trained model would also be beneficial to different, but related tasks, such as facial expression recognition, or gesture recognition [162].

Another direction for the text summarization task could be to expand to other text genres such as public policies to aid reader comprehension, or multi-lingual applications [186]. Transcripts, for example, can be extracted automatically and the proposed summarization systems can be used to highlight some of important sentences or segments. There are many different genres to be explored: policies, contracts, and medical prescriptions. Also, the introduced methods can be employed to

be compared to the ones generated by an abstractive summarization system to evaluate the factual consistency of the abstractive summary.

Finally, given the confluence of different modalities in sequential data, as demonstrated in this dissertation, one could really attempt to literally bring the two worlds together. As we are living in the era of big data, different types of multi-modal and multimedia data are generated every second, e.g. live-streaming videos from numerous streamers, with associated textual and other met-data. One could thus look at joint summarization problems [185], using a common underlying framework, or joint highlights of video and associated text. To conclude, we hope that by emphasizing the common nature of sequential data in this dissertation, we opened new doors to explore different disparate modalities jointly for discovering knowledge from big sequential data.

# LIST OF REFERENCES


[1] *Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset*, 2017.

[2] Jake K. Aggarwal and Lu Xia. Human activity recognition from 3d data: A review. *Pattern Recognition Letters*, 48:70–80, 2014.

[3] Reinald Kim Amplayo and Mirella Lapata. Unsupervised opinion summarization with noising and denoising. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

[4] Nazim Ashraf, Chuan Sun, and Hassan Foroosh. View-invariant action recognition using projective depth. *Journal of Computer Vision and Image Understanding (CVIU)*, 123:41–52, 2014.

[5] Mehala Balamurali and Arman Melkumyan. t-sne based visualisation and clustering of geological domain. In *Neural Information Processing - 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16-21, 2016, Proceedings, Part IV*, pages 565–572, 2016.

[6] Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2011.

[7] Hakan Bilen, Basura Fernando, Efstratios Gavves, and Andrea Vedaldi. Action recognition with dynamic image networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[8] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[9] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[10] Yu Cao, Daniel Barrett, Andrei Barbu, Siddharth Narayanaswamy, Haonan Yu, Aaron Michaux, Yuewei Lin, Sven Dickinson, Jeffrey Mark Siskind, and Song Wang. Recognize human activities from partially observed videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[11] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *CoRR*, abs/1812.08008, 2018.

[12] Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. Faithful to the original: Fact aware neural abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[13] Jaime Carbonell and Jade Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 1998.

[14] Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. Deep communicating agents for abstractive summarization. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

[15] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, , and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and cross-lingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval)*, 2017.

[16] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[17] Yen-Chun Chen and Mohit Bansal. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[18] Yen-Chun Chen and Mohit Bansal. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[19] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. In *Proceedings of ACL*, 2016.

[20] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany, August 2016. Association for Computational Linguistics.

[21] Sangwoo Cho and Hassan Foroosh. Spatio-temporal fusion networks for action recognition. In *Asian Conference on Computer Vision*, pages 347–364. Springer, 2018.

[22] Sangwoo Cho and Hassan Foroosh. A temporal sequence learning for action recognition and prediction. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 352–361. IEEE, 2018.

[23] Sangwoo Cho, Logan Lebanoff, Hassan Foroosh, and Fei Liu. Improving the similarity measure of determinantal point processes for extractive multi-document summarization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[24] Sangwoo Cho, Chen Li, Dong Yu, Hassan Foroosh, and Fei Liu. Multi-document summarization with determinantal point processes and contextualized representations. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 98–103, Hong Kong, China, November 2019. Association for Computational Linguistics.

[25] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy, August 2019. Association for Computational Linguistics.

[26] Ionut Cosmin Duta, Bogdan Ionescu, Kiyoharu Aizawa, and Nicu Sebe. Spatio-temporal vector of locally max pooled features for action recognition in videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[27] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[28] Hoa Trang Dang and Karolina Owczarzak. Overview of the TAC 2008 update summarization task. In *Proceedings of Text Analysis Conference (TAC)*, 2008.

[29] Hal Daumé III and Daniel Marcu. Bayesian query-focused summarization. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2006.

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[31] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018.

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[33] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[34] Ali Diba, Vivek Sharma, and Luc Van Gool. Deep temporal linear encoding networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.

[35] Ying Ding and Jing Jiang. Towards opinion summarization from online forums. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, 2015.

[36] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Trevor Darrell, and Kate Saenko. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 2625–2634, 2015.

[37] Timothy Dozat and Christopher D. Manning. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[38] Yong Du, Yun Fu, and Liang Wang. Skeleton based action recognition with convolutional neural network. In *ACPR*, pages 579–583. IEEE, 2015.

[39] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[40] Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. Learning-based single-document summarization with compression and anaphoricity constraints. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2016.

[41] Ionut C. Duta, Jasper R. R. Uijlings, Tuan A. Nguyen, Kiyoharu Aizawa, Alexander G. Hauptmann, Bogdan Ionescu, and Nicu Sebe. Histograms of motion gradients for real-time video classification. In *2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2016.

[42] Günes Erkan and Dragomir R. Radev. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 2004.

[43] Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In

*Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy, July 2019. Association for Computational Linguistics.

[44] Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy, July 2019. Association for Computational Linguistics.

[45] Christoph Feichtenhofer, Axel Pinz, and Richard Wildes. Spatiotemporal residual networks for video action recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[46] Christoph Feichtenhofer, Axel Pinz, and Richard P Wildes. Spatiotemporal multiplier networks for video action recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[47] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. *CoRR*, abs/1604.06573, 2016.

[48] Basura Fernando, Efstratios Gavves, José Oramas M., Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 5378–5387, 2015.

[49] Basura Fernando, Efstratios Gavves, José Oramas M., Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. pages 5378–5387, 2015.

[50] Basura Fernando, Efstratios Gavves, José Oramas M., Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. In *CVPR*, pages 5378–5387. IEEE Computer Society, 2015.

[51] Katja Filippova, Enrique Alfonseca, Carlos Colmenares, Lukasz Kaiser, and Oriol Vinyals. Sentence compression by deletion with lstms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[52] Dimitrios Galanis and Ion Androutsopoulos. An extractive supervised two-stage method for sentence compression. In *Proceedings of NAACL-HLT*, 2010.

[53] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. Opinosis: A graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2010.

[54] Mike Gartrell, Elvis Dohmatob, and Jon Alberdi. Deep determinantal point processes. *https://arxiv.org/abs/1811.07245*, 2018.

[55] Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[56] Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. Bottom-up abstractive summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[57] Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond T. Ng, and Bita Nejat. Abstractive summarization of product reviews using discourse structure. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[58] Dan Gillick and Benoit Favre. A scalable global model for summarization. In *Proceedings of the NAACL Workshop on Integer Linear Programming for Natural Langauge Processing*, 2009.

[59] Dan Gillick, Benoit Favre, Dilek Hakkani-Tur, Berndt Bohnet, Yang Liu, and Shasha Xie. The ICSI/UTD summarization system at TAC 2009. In *Proceedings of TAC*, 2009.

[60] Rohit Girdhar, Deva Ramanan, Abhinav Gupta, Josef Sivic, and Bryan Russell. ActionVLAD: Learning spatio-temporal aggregation for action classification. In *CVPR*, 2017.

[61] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 580–587, Washington, DC, USA, 2014. IEEE Computer Society.

[62] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2014.

[63] Han Guo, Ramakanth Pasunuru, and Mohit Bansal. Soft, layer-specific multi-task summarization with entailment and question generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[64] Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2009.

[65] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[66] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 961–970, 2015.

[67] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2015.

[68] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[69] Geoffrey Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with EM routing. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[70] Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. Single-document summarization as a tree knapsack problem. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[71] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[72] Kai Hong, John M Conroy, Benoit Favre, Alex Kulesza, Hui Lin, and Ani Nenkova. A repository of state of the art and competitive baseline summaries for generic news summarization. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*, 2014.

[73] J. Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. ParaBank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, 2019.

[74] Dong Huang, Shitong Yao, Yi Wang, and Fernando De la Torre. Sequential max-margin event detectors. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III*, pages 410–424, 2014.

[75] Mohamed E. Hussein, Marwan Torki, Mohammad A. Gowayyed, and Motaz El-Saban. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 2466–2472. AAAI Press, 2013.

[76] Rebecca Hwa. Supervised grammar induction using training data with limited constituent information. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 73–79, College Park, Maryland, USA, June 1999. Association for Computational Linguistics.

[77] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[78] Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception & Psychophysics*, 14(2):201–211, Jun 1973.

[79] Rie Johnson and Tong Zhang. Effective use of word order for text categorization with convolutional neural networks. *CoRR*, abs/1412.1058, 2014.

[80] Michael Kaisser, Marti A. Hearst, and John B. Lowe. Improving search results quality by customizing summary lengths. In *Proceedings of ACL-08: HLT*, pages 701–709, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[81] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *CoRR*, abs/1404.2188, 2014.

[82] Hidetaka Kamigaito, Katsuhiko Hayashi, Tsutomu Hirao, and Masaaki Nagata. Higher-order syntactic attention network for longer sentence compression. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1716–1726, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[83] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 1725–1732, Washington, DC, USA, 2014. IEEE Computer Society.

[84] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017.

[85] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Ahmed Sohel, and Farid Boussaïd. A new representation of skeleton sequences for 3d action recognition. In *CVPR*, pages 4570–4579. IEEE Computer Society, 2017.

[86] Chris Kedzie, Kathleen McKeown, and Hal Daume III. Content selection in deep learning models of summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[87] Gunhee Kim, Leonid Sigal, and Eric P. Xing. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. In *Proceedings of CVPR*, 2014.

[88] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[89] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *CVPR Workshops*, pages 1623–1631. IEEE Computer Society, 2017.

[90] Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.

[91] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[92] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[93] Kevin Knight and Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 2002.

[94] Yu Kong, Dmitry Kit, and Yun Fu. A discriminative model with multiple temporal scales for action prediction. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, pages 596–611, 2014.

[95] Anastassia Kornilova and Vladimir Eidelman. BillSum: A corpus for automatic summarization of US legislation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 48–56, Hong Kong, China, November 2019. Association for Computational Linguistics.

[96] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 1106–1114, 2012.

[97] Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China, November 2019. Association for Computational Linguistics.

[98] Wojciech Kryscinski, Romain Paulus, Caiming Xiong, and Richard Socher. Improving abstraction in text summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[99] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

[100] Alex Kulesza and Ben Taskar. Learning determinantal point processes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2011.

[101] Alex Kulesza and Ben Taskar. *Determinantal Point Processes for Machine Learning*. Now Publishers Inc., 2012.

[102] Philippe Laban, Andrew Hsi, John Canny, and Marti Hearst. The summary loop: Learning to write abstractive summaries without examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

[103] Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. A hierarchical representation for future action prediction. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III*, pages 689–704, 2014.

[104] Zhen-Zhong Lan, Ming Lin, Xuanchong Li, Alexander G. Hauptmann, and Bhiksha Raj. Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

[105] Zhen-Zhong Lan, Yi Zhu, Alexander G. Hauptmann, and Shawn D. Newsam. Deep local video feature for action recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops*, 2017.

[106] Colin Lea, Michael D. Flynn, Rene Vidal, Austin Reiter, and Gregory D. Hager. Temporal convolutional networks for action segmentation and detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[107] Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. Scoring sentence singletons and pairs for abstractive summarization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[108] Logan Lebanoff, Kaiqiang Song, and Fei Liu. Adapting the neural encoder-decoder framework from single to multi-document summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[109] Inwoong Lee, Doyoung Kim, Seoungyoon Kang, and Sanghoon Lee. Ensemble deep learning for skeleton-based action recognition using temporal sliding lstm networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[110] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation. In *IJCAI*, pages 786–792. ijcai.org, 2018.

[111] Chen Li, Fei Liu, Fuliang Weng, and Yang Liu. Document summarization via guided sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[112] Chen Li, Yang Liu, Fei Liu, Lin Zhao, and Fuliang Weng. Improving multi-document summarization by sentence compression based on expanded constituent parse tree. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2014.

[113] Junyi Jessy Li, Kapil Thadani, and Amanda Stent. The role of discourse units in near-extractive summarization. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 137–147, Los Angeles, September 2016. Association for Computational Linguistics.

[114] Chin-Yew Lin. ROUGE: a package for automatic evaluation of summaries. In *Proceedings of ACL Workshop on Text Summarization Branches Out*, 2004.

[115] Hui Lin and Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Proceedings of NAACL*, 2010.

[116] Weiyao Lin, Chongyang Zhang, Ke Lu, Bin Sheng, Jianxin Wu, Bingbing Ni, Xin Liu, and Hongkai Xiong. Action recognition with coarse-to-fine deep feature integration and asynchronous fusion. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[117] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *Proceedings of CVPR*, 2015.

[118] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *ECCV*, 2016.

[119] Mengyuan Liu, Hong Liu, and Chen Chen. Enhanced skeleton visualization for view invariant human action recognition. *Pattern Recognition*, 68:346–362, 2017.

[120] Yang Liu and Mirella Lapata. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy, July 2019. Association for Computational Linguistics.

[121] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *https://arxiv.org/pdf/1907.11692.pdf*, 2019.

[122] Wencan Luo and Diane Litman. Summarizing student responses to reflection prompts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[123] Wencan Luo, Fei Liu, Zitao Liu, and Diane Litman. Automatic summarization of student course feedback. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, 2016.

[124] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. The stanford CoreNLP natural language processing toolkit. In *Proceedings of the Association for Computational Linguistics (ACL) System Demonstrations*, 2014.

[125] Andre F. T. Martins and Noah A. Smith. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the ACL Workshop on Integer Linear Programming for Natural Language Processing*, 2009.

[126] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[127] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[128] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of SIGNLL*, 2016.

[129] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[130] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2018.

[131] Ani Nenkova and Kathleen McKeown. Automatic summarization. *Foundations and Trends in Information Retrieval*, 2011.

[132] Joe Yue-Hei Ng, Matthew J. Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

[133] Joe Yue-Hei Ng, Matthew J. Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. *CoRR*, abs/1503.08909, 2015.

[134] Juan Carlos Niebles, Chih-Wei Chen, and Fei-Fei Li. Modeling temporal structure of decomposable motion segments for activity classification. In *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II*, pages 392–405, 2010.

[135] Juan Carlos Niebles, Chih-Wei Chen, and Fei-Fei Li. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV (2)*, volume 6312 of *Lecture Notes in Computer Science*, pages 392–405. Springer, 2010.

[136] Dan Oneata, Jakob Verbeek, and Cordelia Schmid. The LEAR submission at Thumos 2014, 2014. -.

[137] Paul Over and James Yen. An introduction to DUC-2004. *National Institute of Standards and Technology*, 2004.

[138] Michael Paul, ChengXiang Zhai, and Roxana Girju. Summarizing contrastive viewpoints in opinionated text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 66–76, Cambridge, MA, October 2010. Association for Computational Linguistics.

[139] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[140] Xiaojiang Peng, Limin Wang, Xingxing Wang, and Yu Qiao. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *Computer Vision and Image Understanding*, 150:109–125, 2016.

[141] Xiaojiang Peng, Changqing Zou, Yu Qiao, and Qiang Peng. Action recognition with stacked fisher vectors. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, pages 581–595, 2014.

[142] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[143] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

[144] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[145] Ronald Poppe. A survey on vision-based human action recognition. *Image Vision Comput.*, 28(6):976–990, June 2010.

[146] Luz Rello, Horacio Saggion, and Ricardo Baeza-Yates. Keyword highlighting improves comprehension for people with dyslexia. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*, pages 30–37, Gothenburg, Sweden, April 2014. Association for Computational Linguistics.

[147] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for sentence summarization. In *Proceedings of EMNLP*, 2015.

[148] M. S. Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. *2011 IEEE International Conference on Computer Vision (ICCV 2011)*, 00(undefined):1036–1043, 2011.

[149] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS)*, 2017.

[150] Norman Sadeh, Alessandro Acquisti, Travis D. Breaux, Lorrie Faith Cranor, Aleecia M. McDonald, Joel R. Reidenberg, Noah A. Smith, Fei Liu, N. Cameron Russell, Florian Schaub, and Shomir Wilson. The usable privacy policy project. *Technical Report, CMU-ISR-13-119, Carnegie Mellon University*, 2013.

[151] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.

[152] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[153] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[154] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[155] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[156] Aidean Sharghi, Ali Borji, Chengtao Li, Tianbao Yang, and Boqing Gong. Improving sequential determinantal point processes for supervised video summarization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[157] Chao Shen and Tao Li. Multi-document summarization via the minimum dominating set. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2010.

[158] Yuping Shen and Hassan Foroosh. View invariant action recognition using fundamental ratios. In *Proceedings of CVPR*, 2008.

[159] Yuping Shen and Hassan Foroosh. View invariant recognition of body pose from space-time templates. In *Proceedings of CVPR*, 2008.

[160] Yuping Shen and Hassan Foroosh. View-invariant action recognition from point triplets. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 31:1898–1905, 2009.

[161] Yuping Shen and Hassan Foroosh. Methods for recognizing pose and action of articulated objects with collection of planes in motion, 2014. US Patent 8,755,569.

[162] Chen Shu, Luming Liang, Wenzhang Liang, and Hassan Foroosh. 3d pose tracking with multi-templates warping and sift correspondences. *IEEE Trans. on Circuits and Systems for Video Technology*, 26:2043–2055, 2016.

[163] Chenyang Si, Ya Jing, Wei Wang, Liang Wang, and Tieniu Tan. Skeleton-based action recognition with spatial reasoning and temporal stack learning. In *ECCV (1)*, volume 11205 of *Lecture Notes in Computer Science*, pages 106–121. Springer, 2018.

[164] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *CoRR*, abs/1406.2199, 2014.

[165] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014.

[166] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[167] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1470–1477, October 2003.

[168] Kaiqiang Song, Lin Zhao, and Fei Liu. Structure-infused copy mechanisms for abstractive summarization. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2018.

[169] Sijie Song, Cuiling Lan, Junliang Xing, Wenjun Zeng, and Jiaying Liu. An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In *AAAI Conference on Artificial Intelligence*, pages 4263–4270, 2017.

[170] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012.

[171] Sasha Spala, Franck Dernoncourt, Walter Chang, and Carl Dockhorn. A web-based framework for collecting and assessing highlighted sentences in a document. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 78–81, Santa Fe, New Mexico, August 2018. Association for Computational Linguistics.

[172] Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[173] Chuan Sun, Imran Junejo, and Hassan Foroosh. Action recognition using rank-1 approximation of joint self-similarity volume. In *Proceedings of ICCV*, pages 1007–1012, 2011.

[174] Chuan Sun, Imran Junejo, Marshall Tappen, and Hassan Foroosh. Exploring sparseness and self-similarity for action recognition. *IEEE Transactions on Image Processing*, 24(8):2488–2501, 2015.

[175] Chuan Sun, Marshall Tappen, and Hassan Foroosh. Feature-independent action spotting without human localization. In *Proceedings of CVPR*, 2014.

[176] Shuyang Sun, Zhanghui Kuang, Wanli Ouyang, Lu Sheng, and Wei Zhang. Optical flow guided feature: A fast and robust motion representation for video action recognition. *CoRR*, 2017.

[177] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9, 2015.

[178] Hiroya Takamura and Manabu Okumura. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, 2009.

[179] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[180] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[181] Gongbo Tang, Mathias Muller, Annette Rios, and Rico Sennrich. Why self-attention? A targeted evaluation of neural machine translation architectures. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[182] Kevin D. Tang, Fei-Fei Li, and Daphne Koller. Learning latent temporal structure for complex event detection. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 1250–1257, 2012.

[183] Yansong Tang, Yi Tian, Jiwen Lu, Peiyang Li, and Jie Zhou. Deep progressive reinforcement learning for skeleton-based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[184] Amara Tariq and Hassan Foroosh. Feature-independent context estimation for automatic image annotation. In *Proceedings of CVPR*, 2015.

[185] Amara Tariq and Hassan Foroosh. A context-driven extractive framework for generating realistic image descriptions. *IEEE Transactions on Image Processing*, 26(2):619–632, 2017.

[186] Amara Tariq, Asim Karim, Fernando Gomez, and Hassan Foroosh. Exploiting topical perceptions over multi-lingual text for hashtag suggestion on twitter. In *Proceedings of FLAIRS Conference*, 2013.

[187] Sansiri Tarnpradab, Fei Liu, and Kien A. Hua. Toward extractive summarization of online forum discussions via hierarchical attention networks. In *Proceedings of the 30th Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2017.

[188] Graham W. Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *Computer Vision - ECCV 2010 - 11th European Conference on Computer Vision*, 2010.

[189] Kapil Thadani and Kathleen McKeown. Sentence compression with joint structural inference. In *Proceedings of CoNLL*, 2013.

[190] T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

[191] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 4489–4497, 2015.

[192] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision, ICCV*, 2015.

[193] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek. Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1271–1283, 2010.

[194] Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing and Management*, 43(6):1606–1618, 2007.

[195] Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *CoRR*, abs/1604.04494, 2016.

[196] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 6000–6010, 2017.

[197] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *CVPR*, pages 588–595. IEEE Computer Society, 2014.

[198] Jesse Vig. A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714*, 2019.

[199] G. Vladutz. Natural language text segmentation techniques applied to the automatic compilation of printed subject indexes and for online database access. In *First Conference on Applied Natural Language Processing*, pages 136–142, Santa Monica, California, USA, February 1983. Association for Computational Linguistics.

[200] Heng Wang and Cordelia Schmid. Action Recognition with Improved Trajectories. In *ICCV 2013 - IEEE International Conference on Computer Vision*, pages 3551–3558, Sydney, Australia, December 2013. IEEE.

[201] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 3551–3558, 2013.

[202] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *IEEE International Conference on Computer Vision, ICCV*, 2013.

[203] Hongsong Wang and Liang Wang. Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In *The Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[204] Limin Wang, Yu Qiao, and Xiaoou Tang. Latent hierarchical model of temporal structure for complex activity classification. *IEEE Trans. Image Processing*, 23(2):810–822, 2014.

[205] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4305–4314, 2015.

[206] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *Computer Vision - ECCV 2016 - 14th European Conference*, 2016.

[207] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV (8)*, volume 9912 of *Lecture Notes in Computer Science*, pages 20–36. Springer, 2016.

[208] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Val Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016.

[209] Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. A sentence compression based framework to query-focused multi-document summarization. In *Proceedings of ACL*, 2013.

[210] Min Wang, Baoyuan Liu, and Hassan Foroosh. Design of efficient convolutional layers using single intra-channel convolution, topological subdivisioning and spatial bottleneck. *arXiv preprint*, arXiv:1608.04337, 2016.

[211] Min Wang, Baoyuan Liu, and Hassan Foroosh. Factorized convolutional neural networks. In *Proceedings of ICCV*, pages 545–553, 2017.

[212] Pichao Wang, Wanqing Li, Philip Ogunbona, Jun Wan, and Sergio Escalera. Rgb-d-based human motion recognition with deep learning: A survey. *CoRR*, 2017.

[213] Pichao Wang, Zhaoyang Li, Yonghong Hou, and Wanqing Li. Action recognition based on joint trajectory maps using convolutional neural networks. In *ACM Multimedia*, pages 102–106. ACM, 2016.

[214] Pichao Wang, Zhaoyang Li, Yonghong Hou, and Wanqing Li. Action recognition based on joint trajectory maps using convolutional neural networks. In *ACM Multimedia*, pages 102–106. ACM, 2016.

[215] Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. Actions ˜ transformations. *CoRR*, abs/1512.00795, 2015.

[216] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pages 7794–7803. IEEE Computer Society, 2018.

[217] Y. Wang, M. Long, J. Wang, and P. S. Yu. Spatiotemporal pyramid network for video action recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[218] Daniel Weinland, Rémi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 115(2):224–241, 2011.

[219] Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

[220] Kristian Woodsend and Mirella Lapata. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 565–574, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[221] Yuanjun Xiong, Kai Zhu, Dahua Lin, and Xiaoou Tang. Recognize complex events from static images by fusing deep channels. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1600–1609, 2015.

[222] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018.

[223] Min Yang, Qiang Qu, Ying Shen, Qiao Liu, Wei Zhao, and Jia Zhu. Aspect and sentiment aware abstractive review summarization. *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2018.

[224] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized autoregressive pretraining for language understanding. *https://arxiv.org/abs/1906.08237*, 2019.

[225] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 5753–5763. Curran Associates, Inc., 2019.

[226] Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. Graph-based neural multi-document summarization. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, 2017.

[227] Dani Yogatama, Fei Liu, and Noah A. Smith. Extractive summarization by maximizing semantic volume. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2015.

[228] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l1 optical flow. In *In Ann. Symp. German Association Patt. Recogn*, pages 214–223, 2007.

[229] David Zajic, Bonnie J. Dorr, Jimmy Lin, and Richard Schwartz. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing and Management*, 2007.

[230] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[231] Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In *The IEEE International Conference on Computer Vision ICCV*, pages 2136–2145. IEEE Computer Society, 2017.

[232] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 19(2):4–10, 2012.

[233] Wei Zhao, Haiyun Peng, Steffen Eger, Erik Cambria, and Min Yang. Towards scalable and generalizable capsule network and its NLP applications. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[234] Wei Zhao, Jianbo Ye, Min Yang, Zeyang Lei, Soufei Zhang, and Zhou Zhao. Investigating capsule networks with dynamic routing for text classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[235] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. A C-LSTM neural network for text classification. *CoRR*, abs/1511.08630, 2015.

[236] Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. Selective encoding for abstractive sentence summarization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[237] Wangjiang Zhu, Jie Hu, Gang Sun, Xudong Cao, and Yu Qiao. A key volume mining deep framework for action recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1991–1999, 2016.

[238] Wentao Zhu, Cuiling Lan, Junliang Xing, Wenjun Zeng, Yanghao Li, Li Shen, and Xiaohui Xie. Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks. In *AAAI*, pages 3697–3704. AAAI Press, 2016.