UNIVERSIDADE
BEIRA INTERIOR

Preventing School-bullying through Automated Video Analysis
Versão final após defesa

Edgar Daniel Santos Jesus

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor João Carlos Raposo Neves

Agosto de 2022

Agradecimentos

A realização desta dissertação levou a um enriquecimento profissional e pessoal, derivado das relações pessoais criadas e desenvolvidas ao longo da realização do mesmo. Consequentemente quero começar por dedicar todo este trabalho aos meus familiares que contríbuiram positivamente em momentos de adversidade.
Quero dar um especial carinho a todos os adolescentes que estiveram presentes na realização das filmagens, cruciais para a realização do documento, devido à dedicação e empenho que colocaram nas tarefas propostas, deixando em diversos momentos as suas vidas pessoais.
Um agradecimento a todas as escolas e entidades organizacionais que disponibilizaram recursos e informações importantes para o planeamento e criação de guiões com o intuito de replicar cenários realistas de *bullying*.
Pretendo realçar com enorme gratidão o empenho e apoio teórico e prático, facultado pelo orientador Professor Doutor João Neves, que se mostrou com um elevado espírito crítico e capacidades de ensino e coordenação impecáveis para o planeamento e esclarecimento de dúvidas.
Uma especial atenção ao padrinho, Nuno Pereira, que estando em realização de Doutoramento disponibilizou o seu tempo para facultar ideias e disponbilizar novos conhecimentos, capazes de aumentar drasticamente a corretude e eficiência nas tarefas exigidas ao longo da dissertação.
Por fim, agradeço a todas as pessoas que contribuíram para o meu crescimento profissional e pessoal ao longo da realização deste trabalho.

Resumo

Atualmente, a humanidade luta contra a discriminação, seja ela praticada através de palavras ofensivas ou atitudes violentas. Muitos dos adolescentes que sofrem de *bullying* na escola têm dificuldades no processo de aprendizagem e consequentemente resultados negativos. Os mais recentes estudos feitos por profissionais da área de saúde mostram que o *bullying* pode deixar marcas na vida dos adolescentes através do surgimento de doenças tais como depressão, baixa autoestima, comportamentos auto-destrutivos, entre outras. Obviamente, estes problemas reduzem drasticamente a qualidade de vida da pessoa, uma vez que podem despoletar traumas socais, físicos e psicológicos na vítima. Foram criadas organizações sem fins lucrativos com o intuito de prevenir a ocorrência de ações de bullying nas escolas através de campanhas de sensibilização. Mas para além dessas campanhas, as instituições têm dificuldade em identificar esses acontecimentos, o que impede que se possa dar um correto e rápido suporte à vitima. Estes fatores levam-nos a procurar novas soluções com ajuda de sistemas automáticos, capazes de detetar, no exato momento, a ocorrência de um ato de bullying numa escola e consequentemente as pessoas envolvidas no mesmo.

Com a ajuda de uma associação sem fins lucrativos portuguesa, foi realizado um estudo que procura identificar os comportamentos mais comuns nas pessoas que se encontram envolvidas nestes atos, e os efeitos que podem trazer para a sociedade, com o objetivo de tornar claro os padrões intrínsecos aos atos de bullying, possibilitando desta forma reconhecer com maior facilidade estas ações.

De seguida, foi realizado um estudo aprofundado acerca das tecnologias e ferramentas utilizadas na área de visão computacional e inteligência artificial, que possibilitam a análise de vídeos capturados em câmaras de vigilância, e consequentemente identificam os tipos de ações humanas existentes. Este estudo começa com as abordagens clássicas de aprendizagem profunda, redes neuronais convolucionais 2D e termina com a utilização de redes avançadas onde são implementadas duas redes neuronais convolucionais 3D, cada uma com funções diferentes, uma responsável pela extração de características estáticas e a outra responsável pela análise do movimento.

Antes de se prosseguir para o desenvolvimento, foi realizado um estudo científico em vários trabalhos já efetuados, que abordaram o tema de bullying, no contexo das tecnologias de aprendizagem profunda. Foram encontrados três artigos que estudaram a possibilidade de utilizar diversas arquiteturas de redes convolucionais e diferentes conjuntos de dados para abordar o problema. Com a leitura e análise desses documentos, concluí-se que existe a necessidade de criar um conjunto de dados que caracterizem o problema através de um grande leque de videos com ações de bullying, e a necessidade de desenvolver um modelo que consiga identificar com uma grande taxa de acerto estas ações em vídeos capturados em cenários realistas.

Depois do estudo realizado nos dois capítulos anteriores, foram criados vários guiões para planear cenários encenados de ações de bullying e não-bullying com estudantes em propriedade escolar. As gravções originaram 350 videos, tendo como cenário casas de banho, salas de aula, cantinas e parques exteriores. Outros 200 vídeos foram transferidos da Inter-

net através do site World Star HipHop. Posteriormente, os 550 videos sofreram um processo de limpeza onde foi removido som e as barras pretas presentes nas laterais. O processo de anotação criou vídeos com sequências de tempo entre os 5s e os 12s. O dataset Kinetics 400 também foi transferido e utilizado para os métodos de destilação de conhecimento e ajuste dos pesos com o dataset *YNF*.

Em relação aos modelos utilizados na fase de desenvolvimento, foram implementadas as arquiteturas *SlowFast*, *I3D*, *C2D*, e *FGN*. *FGN* foi o único modelo capaz de convergir para um mínimo quando treinado com pesos incializados aleatoriamente. No final do processo de treino e validação o modelo atingiu uma taxa de acerto no conjunto de teste perto dos 70%, sofrendo uma redução significativa para os 51% quando utilizado o valor de separação ótimo entre as duas classes. Esta redução ocorreu devido à taxa de acerto inicial ter sido calculada com base no valor de separação de 0.5, enquanto que o valor que garante o maior número de verdadeiros positivos e o menor número de falsos positivos é de aproximadamente 0.87.

Uma vez que o conjunto de dados recolhido é de apenas 550 videos, o que implica um reduzido número de instâncias de teste, foi implementada a técnica de treino *K-Fold Cross Validation*, no modelo *FGN*. Este processo atingiu uma taxa de acerto de 65.67%.

Os restantes 3 modelos foram incializados com os pesos do conjunto de dados *Kinetics 400* e sofreram um ajuste dos pesos atráves do processo de treino com o conjunto de dados *YNF*. O facto de estes modelos terem um grande número de parâmetros para atualizar ao longo do treino, implica o uso de grandes conjuntos de dados para convergir para um mínimo quando treinados com pesos inicializados aleatoriamente. O facto de o conjunto de dados recolhido ter apenas 550 vídeos impediu que estes atingissem um bom desempenho quando treinados sem qualquer conhecimento prévio. A arquitetura de rede *SlowFast* atingiu uma taxa de acerto de aproximadamente 83%, quando utilizado o valor de separação entre as duas classes de 0.5. A taxa de acerto no conjunto de teste foi igual quando utilizado o valor ótimo de separação através da métrica *ROC Curve*. O segundo modelo, *I3D* atingiu uma taxa de acerto de 81% no conjunto de teste e quando contabilizado o valor de separação ótimo, aumentou o desempenho para aproximadamente 87%. O último modelo treinado, *C2D* atingiu uma taxa de acerto no conjunto de teste de aproximadamente 77%, acabando por manter a mesma taxa de acerto quando contabilizado o valor ótimo de separação entre classes. Os valores ótimos de separação foram calculados atráves da métrica *ROC Curve*, que procurou o melhor valor de forma a reduzir o número de instâncias falsas positivas e aumentar o número de instâncias verdadeiras positivas.

Em conclusão, este trabalho apresentou um conjunto de dados que expressa várias ações de bullying e não-bullying entre estudantes em propriedade escolar. Este foi criado devido à inexistência de dados que retratem o problema de *bullying* na sua totalidade, para além de violência física, focando-se em situações de gozo, roubo e intimidação. Com o conjunto de dados anotado e limpo, foram utilizados no processo de treino e validação de 5 modelos de aprendizagem profunda para análise de vídeo com o intuito de criar uma aplicação capaz de diferenciar ações de bullying e não-bullying. O modelo que foi capaz de realizar essa distinção com a melhor taxa de acerto foi a arquitetura *I3D*, inicializado com os pesos do conjunto de dados *Kinetics 400*, atingindo 87 % no conjunto de teste, com o valor ótimo de separação

entre classes.
Para trabalho futuro é mencionada a técnica de destilação de conhecimento utilizada para reduzir o tamanho das redes profundas, diminuindo consequentemente os recursos computacionais necessários para executar os modelos. Uma das vantagens do uso desta técnica é a possibilidade de fazer o desenvolvimento de aplicações de inteligência artificial em dispositivos *IoT* com poucos recursos de energia e processamento, mantendo a mesma taxa de acerto adquirida com modelos de maiores dimensões. Devido à sensibilidade da comunidade relativamente ao tema de *bullying* e partilha de dados visuais relativos a crianças menores de idade em escolas, a possibilidade de realizar inferência sem enviar dados pela Internet para grandes data-centers, adiciona uma camada de segurança às aplicações. Outra das sugestões para melhorar o desempenho da aplicação apresentada nesta dissertação é a gravação de novos vídeos, aumentando substancialmente a variedade de ações.

Palavras-chave Classificação de bullying, Visão Computacional, Inteligência Artificial, Análise de Vídeo, Redes Neuronais Convolucionais 3D

Resumo alargado

Capítulo 1 O primeiro capítulo apresenta o contexto e a motivação para a realização deste documento realçando a importância do tema de *bullying* para a sociedade atual, devido às consequências que este tipo de ações têm na vida dos jovens que estão no seu crescimento tanto intelectual como físico, tais como depressões, baixa auto-estima, auto-mutilação, ansiedade e frustação.
Este trabalho foca-se na criação de um conjunto de dados que expressa ações realistas e encenadas de *bullying* em propriedade escolar, através de vídeos. A criação do conjunto de dados é crucial para desenvolver uma aplicação capaz de detetar, através de camâras de vigilância, a existência de casos de bullying nas escolas, com o intuito de advertir uma entidade responsável para intervir, identificar os agressores e agir de acordo com o estabelecido no local de ensino, relativamente à assistência à vitima. De seguida realiza-se um estudo aprofundado na área de aprendizagem profunda, na tarefa de classificação de ações humanas em vídeos, de forma a criar um modelo de inteligência artífical capaz de detetar situações de bullying. A implementação do modelo implica a instalação e manuseio de ferramentas direcionadas à área, treinar, avaliar e testar várias soluções com o objetivo de procurar a solução ótima.
Os objetivos para a realização deste trabalho são:

- Construir um conjunto de dados de vídeos que caracterizem os comportamentos habituais em situações de *bullying*, em propriedade escolar. Estas ações devem incidir especialmente em casos de gozo entre estudantes, roubo de material, intimidação e arremesso de objetos, devido à inexistência destes dados em conjuntos já criados, que só identificam situações violentas;

- Implementar várias soluções de aprendizagem profunda para a classificação de ações de *bullying* em vídeos capturados por cameras de vídeo-vigilância em propriedade escolar. Para a implementação espera-se o treino, validação e teste das arquiteturas com retirada de métricas.

- Conclusões finais com base nos resultados obtidos, apresentando um modelo com os respetivos hiper-paramêtros, capaz de identificar situações de *bullying* com a maior exatidão.

Capítulo 2 O segundo capítulo foca-se em abordar a reunião com a organização sem fins lucrativos *No-Bully*, onde foram mencionados os efeitos que as ações de *bullying* causam na vida dos adolescentes, bem como os típicos comportamentos das vítimas e dos agressores nessas ações. Esta reunião teve como objetivo criar uma base de conhecimento sólida relativa ao *bullying* para criar guiões para as gravações relativas ao conjunto de dados o mais realistas possível. A investigação feita foi essencial para aumentar consideravelmente a taxa de sucesso do modelo de aprendizagem profunda, uma vez que estes são um reflexo dos dados, o que implica que um conjunto de dados conciso tem mais chances de atingir bons resultados no processo de inferência.

A segunda parte do capítulo exibe todos os estudos relativos à área de aprendizagem profunda relacionados com a classificação de ações humanas em vídeos, desde o seus primórdios modelos até às tecnologias e arquiteturas consideradas estado-de-arte nos dias atuais.

Começou-se por estudar as redes convolucionais 2D, na criação de novas arquiteturas em que a dimensão temporal do vídeo era ignorada e a classificação da ação humana era apenas determinada por uma média das classificações obtidas independentemente, em cada uma das frames.

Posteriormente surgiram as redes convolucionais 3D, onde foram implementados filtros 3D capazes de analisar e fazer uma fusão temporal das características de um conjunto de frames referentes a uma determinada ação.

Com base no estudo dos algoritmos clássicos de visão computacional, criaram-se novas arquiteturas de rede divididas em dois fluxos, em que um dos fluxos analisa as propriedades estáticas das frames do vídeo, enquanto que o segundo fluxo, através do cálculo prévio do *optical flow*, identifica padrões nos movimentos existentes num conjunto de frames.

Uma vez que a análise de vídeo implica a análise temporal, foram reestruturadas as arquiteturas de redes utilizadas na tarefa de processamento de linguagens naturais devido às suas propriedades, em que são criados estados que guardam informações relativas as características encontradas em sequências. Estas arquiteturas demonstram resultados positivos mas implicam o uso de grandes recursos computacionais devido à sua impossibilidade de paralelizar.

Derivado dos conhecimentos obtidos pela comunidade cientifica, surgiram os modelos estado-de-arte que foram implementados com base em arquiteturas de rede convolucionais 2D que obtiveram excelentes resultados em classificar imagens. Dessa forma os filtros 2D dessas arquiteturas foram re-implementados de forma a contabilizar a componente temporal, criando a primeira *Resnet 50* 3D. Estas arquiteturas foram posteriormente utilizadas na criação do modelo *SlowFast*, em que duas redes *ResNet 50* 3D foram implementadas, sendo que cada uma delas tem diferentes parâmetros que permitem analisar apenas propriedades estáticas das frames ou características temporais relativas aos movimentos.

A última secção deste capítulo aborda uma técnica que surgiu com o objetivo de reduzir os excessivos recursos computacionais necessários para executar este tipo de arquiteturas. O método de destilação de conhecimento foi inventado com o intuito de transferir conhecimento existente em grandes arquiteturas, com elevado número de parâmetros, para redes com menores dimensões, mantendo praticamente a mesma taxa de acerto. Esta invenção possíbilitou a criação de modelos com alto desempenho capazes de serem executados em dispositivos de reduzido poder computacional, aumentando consideravelmente o nível de segurança destas aplicações, uma vez que os dados não têm que ser enviados pela Internet para servidores de terceiras entidades.

Capítulo 3 Com o intuito de investigar os trabalhos já realizados pela comunidade científica relacionados com o tema de classificação de ações de bullying em escolas, com o apoio de tecnologias de aprendizagem profunda, foram analisados 3 artigos científicos que se focaram em resolver o problema apresentado neste documento.

Preventing School-bullying through Automated Video Analysis

O primeiro artigo científico encontrado procurou resolver o problema através da implementação de uma rede convolucional 2D para extrair características existentes nas frames do vídeo e posteriormente utilizaram uma rede *Long-Short Term Memory* para realizar a fusão temporal das características. Esta implementação deu origem à aplicação em que dois modelos de análise estática e temporal foram incorporados de forma a prever, através de um valor médio, a classe final. Infelizmente, os autores do artigo não disponibilizam nem o conjunto de dados criado nem o código fonte referente às experiências efetuadas.
O segundo artigo abordou o problema através da criação de uma rede convolucinal 3D para análise temporal e estática das características presentes nas frames. Para além da análise temporal também implementaram uma rede para processamento de linguagem natural capaz de classificar ações de bullying através do som produzido pelo ambiente, agressor e vítima. Tal como mencionado no primeiro artigo, nenhum dos conjuntos de dados foram disponibilizados, bem como o código fonte.
Por fim, o útlimo artigo utilizou uma abordagem totalmente diferente, em que foram determinados através de algoritmos clássicos de visão computacional, as posições do esqueleto dos participantes em cada frame, de forma a classificar ações de bullying com suporte a uma rede convolucional de grafos. O conjunto de dados utilizados na implementação deste artigo baseou-se na re-criação de ações encenadas de bullying entre estudantes em propriedade escolar, mas devido à sensibilidade do tema não foram disponíbilizados ao público.

Capítulo 4 Para a implementação deste trabalho foram utilizados dois conjuntos de dados, sendo que o primeiro *Kinetics 400* tem como objetivo ser utilizado em métodos de *fine-tuning* e *Knowledge Distillation*. O segundo é crucial para o desenvolvimento desta dissertação e teve um grande impacto na realização do mesmo, sendo que foram criados guiões com o intuito de criar cenários encenados com estudantes. Foram realizados quatro dias de gravações onde foram adquiridos 350 vídeos de bullying e não-bullying em escolas regionais. Outros 200 vídeos foram transferidos da Internet através do site *WorldStarHipHop*, com o objetivo de criar um conjunto de dados o mais realista possível. Todos os dados foram limpos e devidamente anotados, criando um conjunto de dados final de 550 vídeos, com sequências temporais entre os 5 e os 12 segundos. Este processo deu origem ao conjunto de dados *You Never Forget*.

Capítulo 5 O capítulo 5 é direcionado à implementação de 5 modelos diferentes, *FGN*, *SlowFast*, *I3D*, *C2D* e *Knowledge Distillation FGN*. Para o primeiro modelo *FGN* foi utilizada a biblioteca *Keras* para implementar uma função capaz de carregar os dados existentes em ficheiros *Numpy*, com as frames relativas aos vídeos e os correspondentes *optical flows*. O modelo foi implementado com as funções existentes no *Keras*, bem como todo o processo de treino, validação e teste da respetiva arquitetura. O processo de implementação deste modelo implicou a realização de várias experiências de forma a procurar os hiper-parêmetros ótimos. Os modelos *C2D*, *I3D* e *SlowFast* foram implementados tendo como suporte a biblioteca *Pytorch*. Para o carregamento de dados foi implementado um ficheiro *Python* capaz de estender as funções já existentes na biblioteca, criando desta forma funções customizadas. Estes mod-

elos contêm um ficheiro de configuração que torna o procedimento de desenvolvimento mais versátil, possibilitando alterar as variáveis relativas à aplicação sem necessidade de modificar o código. Em cada ficheiro de configuração, relativo a cada modelo, foram adicionadas variáveis de controlo de treino, validação e teste, tais como número de épocas, taxa de aprendizagem, ficheiros para guardar métricas e criação de gráficos. Os modelos foram treinados sem qualquer conhecimento prévio derivado de outros conjuntos de dados e também tendo como base os pesos do conjunto de dados *Kinetics 400*.

O último modelo *Knowledge Distillation FGN* foi implementado com o intuito de transferir o conhecimento adquirido pelos modelos *C2D*, *I3D* ou *SlowFast* devido à inexistência de pesos relativos ao modelo *FGN* no conjunto de dados *Kinetics 400*. Este processo teve como objetivo aumentar a taxa de acerto do modelo *FGN* através da possibilidade de realizar o processo de *Fine-tuning* com o conjunto de dados *You Never Forget*. Desta forma, tentou-se criar um modelo com alto desempenho na tarefa de classificar ações de bullying, com reduzido número de parâmetros, e consequentemente possibilidade de ser executado em equipamentos de reduzido poder computacional.

Capítulo 6 O capítulo 6 é crucial nas conclusões finais desta dissertação, uma vez que apresenta todos os resultados obtivos durante o desenvolvimento dos vários modelos de aprendizagem profunda.

Começando pelo modelo *FGN*, este foi o único capaz de convergir para um mínimo através do processo de otimização, quando treinado sem qualquer conhecimento prêvio. Este facto deu-se devido ao número de parâmetros para atualizar ao longo do treino, sendo este valor de 272.690 mil paramêtros. Os restantes modelos apresentam um valor a rondar os milhões de paramêtros o que implica a utilização de conjuntos de dados com elevado número de instâncias, para atingir uma solução ótima sem qualquer conhecimento prêvio. O modelo *FGN* atingiu uma taxa de acerto a rondar os 70% no conjunto de teste, quando utilizado um número de frames de 4 e um valor de separação entre as duas classes de 0.5 .

Os modelos *C2D*, *I3D* e *SlowFast* apenas obtiveram resultados quando treinados com os pesos do conjunto de dados *Kinetics 400*. A arquitetura *SlowFast* atingiu uma taxa de acerto no conjunto de teste a rondar os 83%, quando contabilizado um número de frames de 2. O modelo *I3D* por sua vez atingiu o valor de taxa de acerto no conjunto de teste de aproximadamente 81%. Por fim, o modelo *C2D* adquiriu uma taxa de acerto no conjunto de teste de aproximadamente 77%. Todos os resultados obtidos até esta fase de desenvolvimento foram obtidos tendo a inferência do conjunto de dados de teste sido realizada com valor de separação entre as duas classes de 0.5, o que implica que não se pode retirar conclusões 100% concisas relativas aos desempenhos dos modelos. Para isso foi feito um último estudo aos modelos, onde foram determinados os valores ótimos que separam as duas classes de forma a aumentar o número de instâncias verdadeiras positivas e consequentemente reduzir o número de falsas positivas. Depois deste processo foi possível identificar que o modelo com melhores resultados foi o *I3D* com uma taxa de acerto de aproximadamente 87% no conjunto de teste, tendo o valor de separação ótimo sido calculado com base nos dados de treino.

Capítulo 7 As conclusões de todo o trabalho encontram-se no capítulo 7, onde são apresentadas ideias de melhoramento dos métodos implementados ao longo deste documento, tais como o uso de *Transformers* para realizar a classificação de ações humanas em vídeos, uma vez que têm demonstrado bons resultados na resolução de problemas ligados à área de visão computacional. Outra das melhorias sugeridas é a gravação de vídeos de *bullying*, com o intuito de aumentar a diversidade das ações apresentadas no conjunto de dados *YNF*. As últimas sugestões de melhoria estão relacionadas com parte do trabalho de desenvolvimento neste documento, *Knowledge Distillation*, uma vez que a sensibilidade do tema de bullying implica a criação de aplicações extremamente seguras, capazes de preservar os dados capturados nas escolas, podendo esta técnica ser crucial para a aceitação deste tipo de abordagens no tratamento de casos de *bullying* nas escolas.
Este trabalho apresentou um conjunto de dados de vídeos que retratam ações de bullying e não-bullying por parte de jovens em propriedades escolares, sendo que as ações mais presentes são a intimidação, roubo de objetos, ameaça e arremesso de objetos, uma vez que não existia nenhum conjunto de dados com estas características. Com base no conjunto de dados formulado *YNF*, consegui-se implementar um modelo de aprendizagem profunda capaz de classificar vídeos de *bullying* com uma taxa de acerto no conjunto de teste de 87%, sendo este valor bastante fundamentado com o cálculo de todas as métricas necessárias ao desenvolvimento de soluções com base nas tecnologias e métodos da área de aprendizagem profunda.
Infelizmente, não se conseguiu obter resultados na implementação do modelo *Knowledge Distillation FGN*, devido aos recursos temporais e de equipamento necessários para a correta execução do mesmo, sendo necessário um disco rígido com capacidade superior aos *10 TB*, e um ciclo de 15 dias para executar uma experiência. Ficando este modelo implementado e pronto para ser trabalhado em situações de melhoria futuras.

Abstract

Currently, humanity strives to prevent discrimination, whether through offensive words or violent attitudes. Most teenagers who suffer bullying in school have difficulties in the learning process and consequently low grades. Most of the recent studies carried out by professionals in the health department show that the marks left by events of this type can bring illnesses such as depression, low self-esteem, and self-destructive behaviors. To address this problem non-profit institutions appear to prevent this kind of action through sensibility campaigns. However, these institutions have limitations that make it impossible to diagnose most of these occurrences, creating a lack of assistance for the victim. These reasons motivate us to search for new solutions with the help of automated systems that will make it possible to detect, at the exact moment, the persons involved in bullying actions in school property.

With the help of a Portuguese non-profit bullying organization, a study was made to collect information about the most known behaviors of persons involved in bullying actions and their effects on society to have good guidelines to identify this events. Next, we carried out an investigation about technologies used in computer vision and artificial intelligence that allow the analysis of videos captured by surveillance cameras and can predict which type of action is inhered in each one. We present a variety of architectures since the first model capable to classify human behavior on videos, until the current times, where state-of-the art architectures, composed by two 3D convolutions streams, able to extract spatial and temporal features were developed.

To search previous studies in the deep learning area related to bullying recognition in school videos, three scientific papers were found that already had investigated this kind of problem. Our analysis derived by the studies shows us the need to create a novel dataset able to represent all types of existing bullying actions and a new model architecture capable of identifying these events with high accuracy.

Following the previous studies made in Chapter 2 and 3, a few guidelines were created to mimic bullying behavior on school grounds with a group of teenagers. Three hundred fifty clips were shot in bathrooms, classrooms, hallways, and canteens with five kids aged 7 to 18 years old. Another 200 films were acquired from the Internet and categorized alongside the recorded videos, producing a balanced dataset of 550 trimmed videos. The data cleaning process removed audio and black sidebars. The Kinetics 400 was downloaded and applied for fine-tuning deep learning pipelines.

In terms of models, the *SlowFast*, *I3D*, *C2D*, and *FGN* architectures were used to construct the application. The *FGN* was the only model that produced plausible results when trained from scratch, finishing the training process with an accuracy on the test dataset of around 70%. However, when the ideal threshold is employed, this value drops to around 51%. Following the successful training from scratch with the *FGN*, a training strategy known as K-Fold Cross Validation was implemented, which divided the dataset into ten pieces to test the entire dataset. The final result is the average of the ten models, which attained an accuracy of 65.67%.

When trained from scratch, the other three models could not converge to a minimum and only got satisfying performance when fine-tuned using the *Kinetics 400* weights. These three models do not perform well when trained from scratch since they contain numerous parameters that must be changed, signaling that more extensive datasets are required. The *SlowFast* model obtained approximately 83% when selecting the class with highest probability. However, this score was maintained when adopting the optimum threshold. The *I3D* model scored 81% on the test dataset, when considered the class with highest probability. However, determining the appropriate threshold achieved the best accuracy of approximately 87%. Finally, the *C2D* model obtained approximately 77% accuracy on the test dataset. This model maintained this performance when computed and utilizing the optimum threshold. These thresholds were determined using the *ROC Curve*, which looked for the best threshold with the highest number of true positives and the lowest amount of false positives.

Ultimately, this study offered a unique bullying dataset with activities that highlight the bullying theme and have more attributes than well-known conflict datasets. After cleaning and labeling the dataset, 550 bullying and non-bullying trimming films were produced. Due to the sensitivity of the topic and the requirement for authorization from the student's responsible entity, the filming procedure of the movies, getting the school locations and students, was challenging.

It was suggested for future work to use network compression techniques through knowledge distillation, teaching a student model with a smaller size with knowledge derived from a huge model, to reduce the number of parameters and thus the number of computing resources while maintaining accuracy. This approach has advantages since it allows the model to be performed in inference mode on IoT devices rather than transferring data over the Internet to large data centers. This method provides an additional security layer to an application because of the sensitive bullying topic and school video information. Another enhancement proposal is to record new bullying and non-bullying films to offer more features and variation to the dataset.

Keywords Bullying Classification, Computer Vision, Artificial Intelligence, Video Analysis, 3D CNNs

Contents

List of Figures

List of Tables

Lista de Acrónimos

2S-AGCN	Two-Stream Adaptive Graph Convolutional Network
2s-AAGCN	Two-Stream Attention Adaptive Graph Convolutional Network
AAGCN	Attention Adaptative Graph Convolutional Network
B-GCN	Bone Graph Convolutional Network
CNN	Convolution Neural Network
CSV	Comma-separated Values
IoT	Internet-of-Things
GCN	Graph Convolutional Network
J-GCN	Joint Graph Convolutional Network
LSTM	Long Short-Term Memory
MFCC	Mel-Frequency Cepstral Coefficients
OCR	Optical Character Recognition
RNN	Recurrent Neural Network
ROC	Receiving Operating Characteristic
SOTA	State-of-the-Art
ST-GCN	Spatial Temporal Graph Convolutional Network
UBI	Universidade da Beira Interior

Chapter 1

Introduction

This chapter presents the context and motivations carried in this dissertation. The objectives mentioned in the second section show the reader the main issues that we will address. At the end, there is a section referring to the main contributions that this document can offer to society and an outline describing the material addressed in each chapter.

1.1 Context and Motivation

This dissertation was proposed for a master's degree in computer science at UBI and will address the recognition and prevention of bullying actions presented in school's properties through videos recorded with surveillance cameras.
A study will be done to search deep learning technologies to classify bullying actions in videos through CNNs able to extract temporal and static features from each frame.
This dissertation's motivation is to record a novel dataset based on bullying actions in school property and train, fine-tune, and test deep learning models addressing the task of action recognition using the data recorded and annotated.

1.2 Objectives

The major objective for this dissertation is the development of a solution to identify bullying actions in real-world situations through surveillance cameras. For this, it will be necessary to capture videos showing bullying events in school property, staged and real, and annotate them to the task of action classification. After the acquisition data, it will be used to train, test, and fine-tune a variety of deep learning models to determine which one is the must accurate to create a real-world application.
Also, it is necessary to make a study on the existing deep learning technologies used in computer vision by the scientific community to analyze video and extract features allowing the classification of different types of actions.
At the end, the contribution to society is a piece of intelligent equipment that can process and tell if a chunk of a video contains a bullying act and reach someone responsible for addressing these situations.

1.3 Dissertation Outline

This dissertation is organized as follows:

- **Chapter 2**: The second chapter presents background material where the deep learning approaches used in video analysis prevention, from the first methodologies used to classify human activity in videos to the SOTA methods. It shows a side view about bullying from professionals who work in a non-bully organization;
- **Chapter 3**: Presents the studies made by the scientific community about technologies and methodologies used in deep learning to the task of bullying classification;
- **Chapter 4**: This chapter is focused in the datasets used in this document, with special attention to the *Never You Forget* dataset recorded and annotated with the purpose for creating a dataset with novel types of behaviors.
- **Chapter 5**: The development phase is explained in the fifth chapter and is crucial to the reader be able to understand the practical side of the technologies used in this project.
- **Chapter 6**: The sixth chapter has a very important role showing the deep learning train, validation and test plots and tables metrics.
- **Chapter 7**: The last chapter is dedicated to the conclusions achieved through the technologies used and presents a model capable to detect bullying . At the end, a section is used to mention further improvements for the research community.

Chapter 2

Theoretical Background

2.1 Bullying Information Gathering

This section describes the information acquired in the online meeting with a non-bully organization, where topics such as behaviors of the victims and aggressors in bully situations were discussed, and the main physical and psychological effects in the society.

2.1.1 No-Bully Organization

An online meeting was made with the Portuguese *No-Bully* organization to retrieve information about the victims and bullies' behaviors when involved in bully actions. Some aspects were presented as a list of the most well known actions made by the aggressors when they approach the victims, like stealing objects, throwing objects and liquids, making them pursue things that were stolen by the aggressor, making them remove clothes, or do activities against their will, and in some cases the use of physical violence.
Besides the list of actions, some of the causes of the aggressor's actions were mentioned, like past situations with violence or bullying events that have made them people with social difficulties.
Another aspect taken into account in the meeting were the actions' effects on the victims. Some are psychological, like depression, suicide, anger, fear of going back to curricular activities, and low grades. In terms of violent bullying, some victims have scars on their faces or bodies, making them uncomfortable with their appearance. This information is relevant for this project because it shows the importance of preventing bullying in schools and supports the foundations of the staged actions performed for the data captured using a surveillance camera in school proprieties, allowing the construction of a video dataset needed for training the deep learning models.

2.2 Deep Learning Approaches for Video Understanding

This section was created to give information to the reader about the deep learning methodologies used in video analysis. The information provided in the next section is crucial for understanding further studies made in this document. First, it describes the pioneer strategies devised for frame sequence analysis that allowed the creation of video classification methods. Then, it describes the advances made to these strategies through the use of biological inspired mechanisms. At the end, it explains a SOTA method, created by the Facebook Research team, SlowFast, that shows to be one of the best architectures in terms of accuracy and time of inference.

2.2.1 2D Convolutions

The deep learning area allows us to use neural networks to process a group of frames, representing for the first time a new temporal dimension. A video has two components, the spatial in two dimensions and the temporal in one dimension. This definition gave information to the research team and allowed them to create the first deep neural network that could process a video and return predictions about action recognition, detection, segmentation, and tracking of objects and people on videos.

The first architecture was based on a conventional approach using a standard 2D convolution for each frame and returned an average of all the predictions. This approach, in the beginning, appeared to be inefficient, but the results showed good performance.

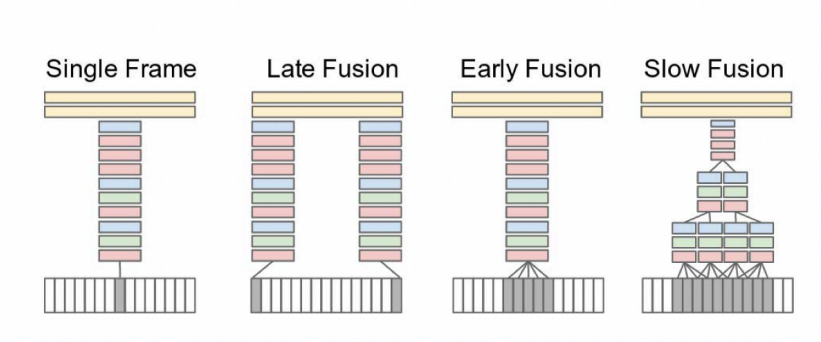

Figure 2.1: Illustration of the fusion process incorporated in the neural network architectures. Image from [4].

Figure 2.1 shows the different types of fusion that were implemented, allowing to average the predictions returned by the neural network. The first is called single frame and, as mentioned before, is just a 2D CNN that processes every single frame independently and average the results. The next one is the late fusion that uses more than one 2D CNN and a dense layer at the end of the networks to fuse all the predictions through an automatic process. Early fusion was the designation given when the fusion occurred in the first step. This model fuses the frames before the processing phase of the 2D CNN. The last architecture is slow fusion, also known as 3D Convolutions.

2.2.2 3D Convolutions

As presented before, there are four pipelines to process video. Some use 2D Convolutions, fusing the spatial and temporal features at different stages. A 3D convolution approach called Slow Fusion was also created because the extraction of spatial and temporal features occur slowly in the progression of the network, as illustrated in Figure 2.2

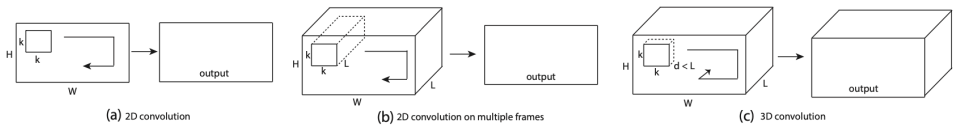

Figure 2.2: Illustration of existing approaches to process video compared with traditional 2D Convolutions. **(a)** Representation of traditional 2D convolutions on images, the resulting process returns a image. **(b)** 2D approach with temporal dimension, also resulting in a image. **(c)** 3D convolution approach with spatial and temporal dimensions able to slowly extract features from both dimensions. The result is a cube matrix with time information. Image from [5].

With the existing 3D convolutions approaches, a C3D network was created with similar 2D convolutions architectures but could process temporal dimensions.

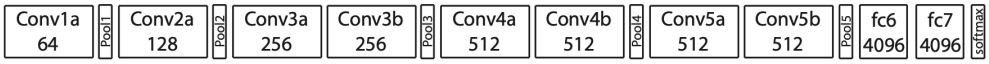

Figure 2.3: C3D network architecture. Image from [5].

Figure 2.3 shows the C3D network architecture, which is composed of eight 3D Convolutions, each one followed by a Pool layer and in the end with two dense layers and a softmax layer. All of the 3D convolutions have 3x3x3 kernel size and the pool layers 2x2x2 kernel size, except the first one with 1x2x2 kernel size.

2.2.3 Two-Stream Network

Some studies about how humans see the world showed us that we process motion and static information differently, making us able to recognize actions only with movement. The problem with the 3D Convolutions is that both spatial and temporal dimensions are treated the same way. This information brought a new deep learning approach trying to solve the task of video action recognition separating both pathways.

Optical Flow Optical flow is an algorithm capable of detecting motion from two consecutive video frames. The process can compute a flow field or distortion field with vector displacement that tells where each one of the pixels will move to in the next frame. The algorithm's execution returns a matrix with each position, an x and y component of the pixel movement. Figure 2.4 shows the optical flow process described before.

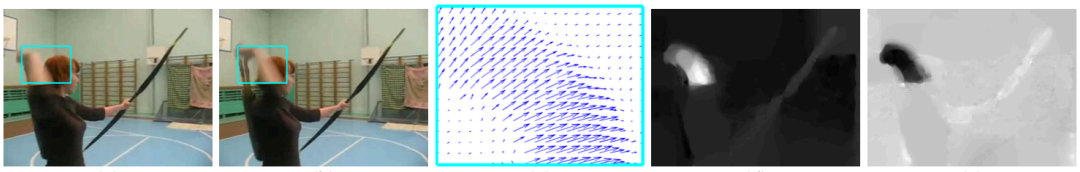

Figure 2.4: Optical Flow algorithm process. **(a) and (b)** Pair of consecutive images from a video with a respective area annotated. **(c)** Algorithm's execution returns a matrix with vectors containing information about the movement of each pixel in the images. **d)** Image with respect to the movements in the horizontal x axis. **(e)** Image with respect to the movements in the vertical y axis. Images from [6].

Optical flow is used to create a two-stream network allowing it to focus on spatial features and temporal features through motion. The first architecture was built with two parallel convolutions networks stack, one to classify each video frame and the second to analyze the motion extracted by optical flow.

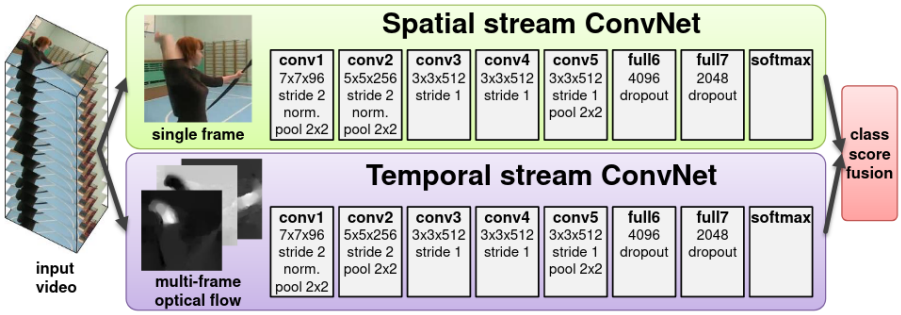

Figure 2.5: Two-stream network architecture for video classification. Image from [6].

Figure 2.5 shows the components used in two-stream networks. As mentioned before, there are two convolutional networks. The first spatial stream, ConvNet, processes each frame of the sequence and has five 2D convolutions and two fully-connected layers with Softmax at the end. The second convolution network is fed with motion information extracted by the optical flow algorithm and fused with the early fusion method. The number and type of layers are the same as the first one. In test time, the results are the average of the probability distribution of each stream.

2.2.4 Long-Term Recurrent Convolution Network

Until now, the previously presented architectures could only explore a local receptive field on the extraction of temporal features, making it challenging to process motion sequences with long structures (e.g.,more than 5 seconds). With the need to process longer frame sequences, the researchers created a new model for action classification with an already existing model used in natural language processing known as RNN.

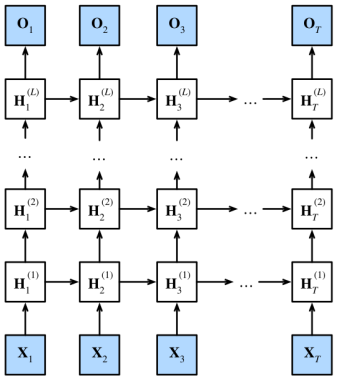

Figure 2.6: Deep Recurrent Neural Network architecture. Image from book [7].

Natural language processing created a deep recurrent neural network 2.6 composed of more

than one hidden layer for processing long text sequences. This model functionality can be described as continuous information passing between the early hidden states and the next ones of the current time step layer and the current time step of the next layer.

The authors at [6] proposed a novel deep architecture to process long sequences of frames with the same ideology used in the deep recurrent neural network but replaced the input of the deep recurrent neural network with 2D or 3D convolutions. This change allowed the fusion of the features extracted by the convolutions along the time.

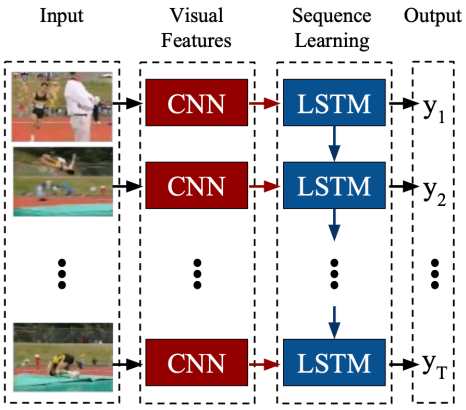

Figure 2.7: Long-Term Recurrent Convolution Network architecture, with 2D or 3D CNNs to extract images features and LSTMs to fuse temporal information. Image from [8].

Figure 2.7 illustrates a network architecture built with 2D or 3D CNNs and LSTMs. The input to the CNN is each one of the frame sequences, and after the feature extraction process, the LSTMs perform the features fusing process, making possible the action recognition of long sequences. The authors chose LSTMs because of their nonexistent vanishing problems.

2.2.5 Spatio Temporal Self-Attention

RNNs architectures were created to process long sequences, but on their first stage, they had gradient vanishing problems because the information was contained in only one single state, known as a context variable. This context variable was used to give information to the decoder about the sequence features extracted by the encoder.

Attention Attention, presented in Figure 2.8, appeared as a mechanism that solved the gradient vanishing and non-parallelization problems of the RNN. Attention computes not only one context vector containing all the sequence information but also computes one context vector for each state of the RNN. This mechanism is foundations based on the functionality of the human eye because we focus only our vision on a specific target.

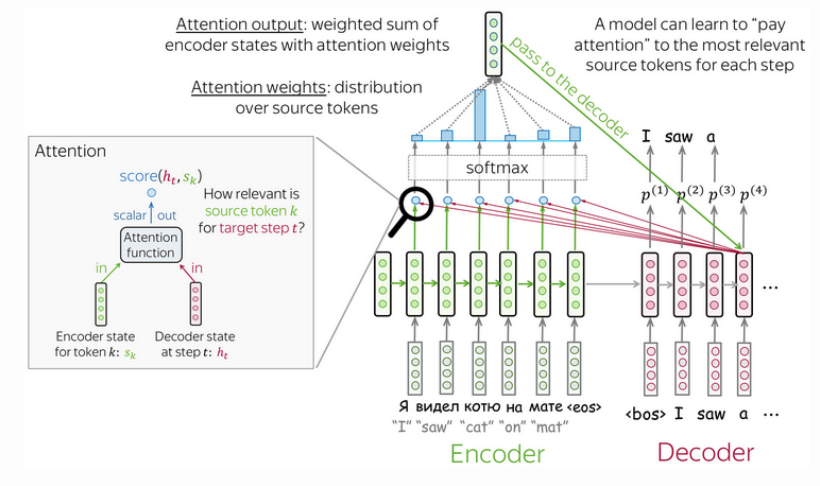

Figure 2.8: Attention computed in the task of natural language processing with RNNs. Image from [9].

Self-Attention Self-Attention, presented in Figure 2.9, is a layer that uses attention but computes it differently. It uses a query, value, and key matrix to compute the attention weight layer and, from it, learns the most relevant features. These self-attention layers are stacked together, constructing one block known as multi-head self-attention.

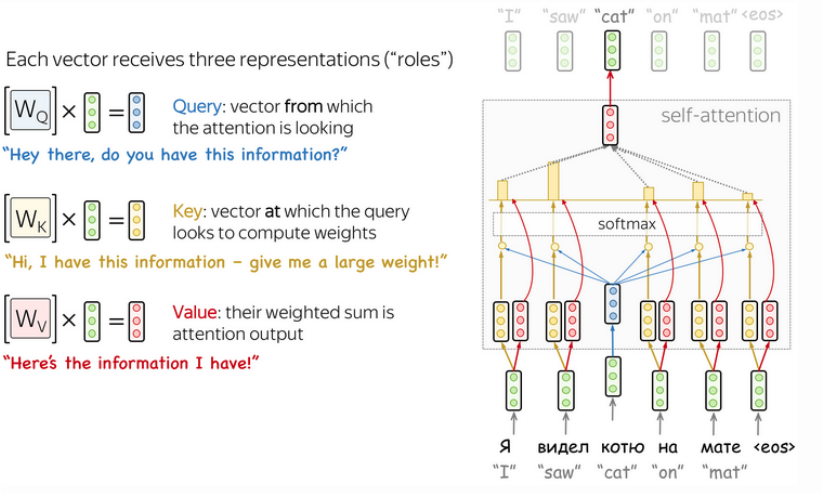

Figure 2.9: Self-Attention computed in the task of natural language processing with RNNs. Image from [9].

In the case of processing videos, an architecture was created that uses 3D convolutions to extract spatial and temporal features from the sequence of frames, mixed with self-attention layer in between these convolutions to achieve global temporal features.

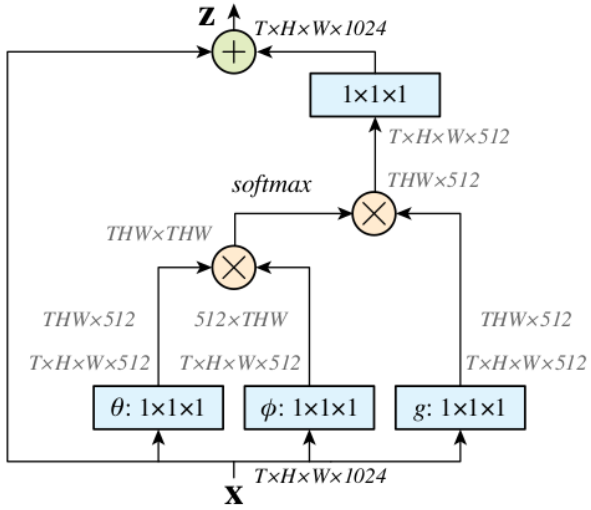

Figure 2.10: NonLocal Block with attention used in between 3D convolutions to fuse temporal information. Image from [10].

Figure 2.10 shows a self-attention layer used to process long sequences of frames and search which features are relevant for each action recognition, called NonLocal Block. These blocks are used between 3D convolutions and built a model 3D CNN with attention to fuse temporal information. This architecture brought outstanding results in the action recognition task, but it is crucial to choose the best 3D CNN architecture to achieve better performance to extract the spatial and temporal features.

2.2.6 Inflating 2D Networks to 3D (I3D)

The task of image classification through 2D CNN architectures has been a very researched topic in the last years. The authors at [11] used the Inception 2D CNN architecture known as InceptionV1 and changed it, raising the number of channels. This way, a new 3D CNN 2.11 was created with a base of a 2D CNN with excellent known results for image classification tasks.

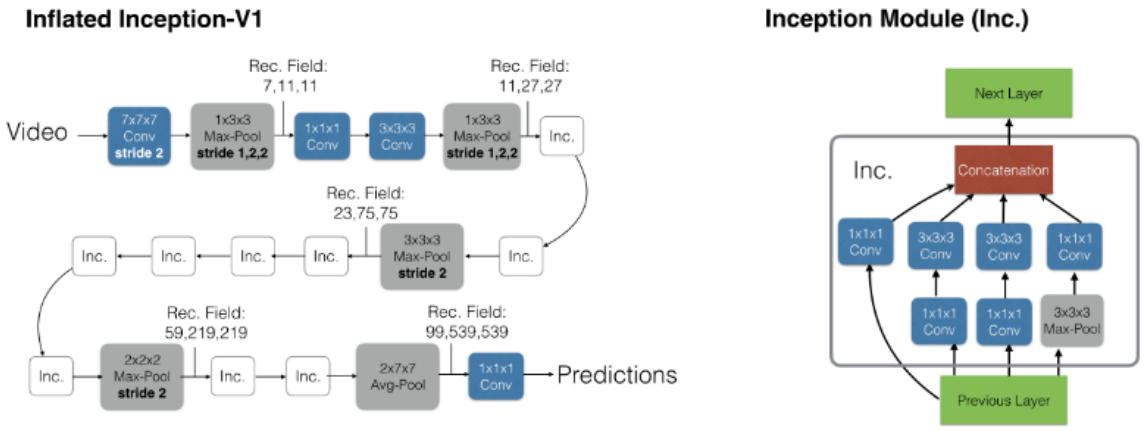

Figure 2.11: 2D CNN model InceptionV1 inflated to 3D CNN. Image from [11].

2.2.7 SlowFast Network

As mentioned at 2.2.6, new networks have been created for video processing through the process of inflating well-known 2D CNN networks like VGG, Restnet, and others, creating 3D CNNs with the ability to analyze temporal features.

The SlowFast network was built taking into account the primary function of the human eye, using two different paths to analyze various features. P-Cells that operate at a high frame rate, allowing us to analyze the movement and M-cells with a low frame rate that focuses on capturing features about colors, shape, and distance of objects/persons.

The fusion of the inflation technique and the human eye functionality created the SlowFast network 2.12 with two resnet 50 or 100 inflated to 3D CNN, one of them is called slow path because it has a low frame rate but a significant number of channels to process, with colossal detail, all the features of each frame. The other is called the fast path because it has a higher frame rate but a small number of channels, focusing mainly on extracting temporal features.

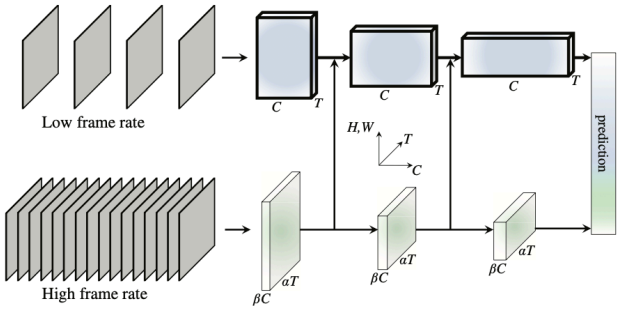

Figure 2.12: SlowFast network architecture shows two paths, slow composed with a low frame rate and a high number of channels and fast built with a high frame rate and a low frame rate. The lateral connection allows exchanging features between the two paths. Image from [12].

One of the main goals of creating the SlowFast network was to build an end-to-end trainable network without external algorithms like optical flow used in previous deep learning models for video understanding.

A comparative study was made at [12] against the most known models trained with Kinect400 and Kinect600 datasets, and the results showed that it is a lot superior in terms of top1 and top5 accuracy metrics.

2.2.8 Flow Gated Network

Cheng et al. [13] created a novel dataset related with real-world fights captured by surveillance cameras. In the learning process, they built a model using three 3D CNN, the first for RGB feature extraction, the second for optical flow, and the third for merging information from the output of the self-learning temporal pooling.

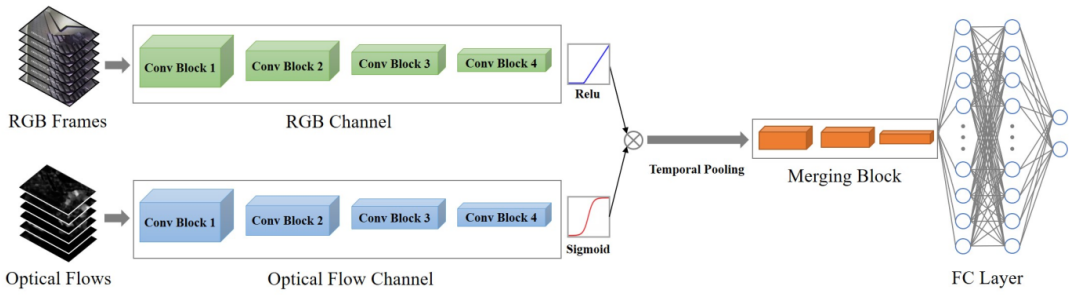

Figure 2.13: The structure of the Flow Gated Network. Image from [13].

Figure 2.13 shows the blocks used on the architecture of the Flow Gated Network, and it is possible to analyze the output of the first two 3D CNN. The first is responsible for extracting spatial features from the RGB frames and, in the end, has a Relu activation function. The second 3D CNN processes the optical flow frames and has a Sigmoid activation fuction. The next phase is the multiplication process of the output of each one of the previous paths. The sigmoid function only returns values between 0 and 1, which means that after the multiplication process and maximum temporal pooling mechanism, the features are focused only on the movement scene. This process is known as a cropping strategy using dense optical flow 2.14.

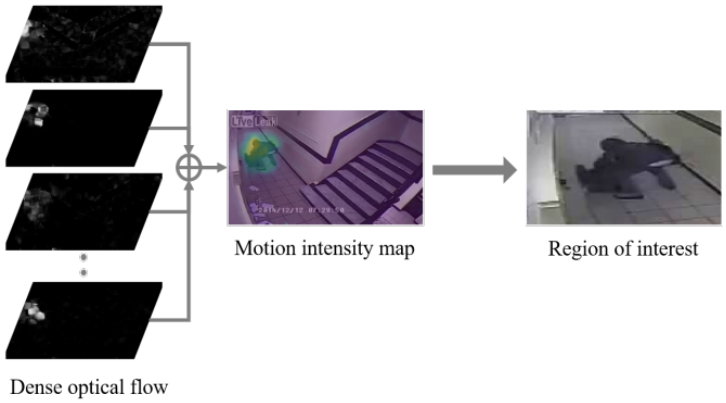

Figure 2.14: Cropping strategy using dense optical flow. Image from [13].

The next block of this model is a 3D CNN merging block which only will extract features from the region of interest. At the end, there is a fully-connected layer responsible for classification.

Block Name	Type	Filter Shape	T
RGB/Flow Channels	Conv3d	1×3×3@16	2
	Conv3d	3×1×1@16	
	MaxPool3d	1×2×2	
	Conv3d	1×3×3@32	2
	Conv3d	3×1×1@32	
	MaxPool3d	1×2×2	
Fusion and Pooling	Multiply	None	1
	MaxPool3d	8×1×1	1
Merging Block	Conv3d	1×3×3@64	2
	Conv3d	3×1×1@64	
	MaxPool3d	2×2×2	
	Conv3d	1×3×3@128	1
	Conv3d	3×1×1@128	
	MaxPool3d	2×2×2	
Fully-connected Layers	FC layer	128	2
	Softmax	2	1

Figure 2.15: Parameters of the model architecture, (The T represents the number of repeats). Image from [12].

Figure 2.15 presents the topology of the network, with the filters shapes and number of feature maps. The authors also used the MobileNet concepts, depth-wise separable convolutions, and pseudo-3D convolutions layers to reduce the model parameters significantly without performance loss.

2.3 Knowledge Distillation

Knowledge Distillation emerged as a method for training a small network to replicate the behavior of a larger network or an ensemble model. Big models with numerous parameters trained on datasets with thousands of terabytes of data produced the most outstanding results in all tasks in deep learning. The most remarkable results in deep learning contests were also obtained using an ensemble model. Because of the resources required for both stages, training, and inference, these constraints make it challenging to implement this model in real-world applications.

Knowledge Distillation is a strategy that attempts to tackle these challenges by compressing networks. A student model is created by distilling the knowledge of a cumbersome or ensemble model. Because the student network is smaller than the teacher network, it requires less training time, fewer resources, and quicker inference time in deployment. This technique minimizes the computer resources required to construct deep learning applications. This method improves the security of deep learning applications by allowing all inference procedures to be performed on IoT devices rather than sending critical data over the internet to massive data centers.

The authors at [14] discovered that their procedure is only achievable due to the usage of soft targeting labels, presented in Figure 2.16, which provide more information in the training process than hard target labels. This finding is termed dark knowledge [15] of the network, as an analog of the existent dark matter in the Universe, which is impossible to perceive and is usually overlooked. However, it has a large global influence. The small student neural networks learn this dark information by learning the relative distribution of the soft label

targets acquired by the final softmax label of the teacher model.

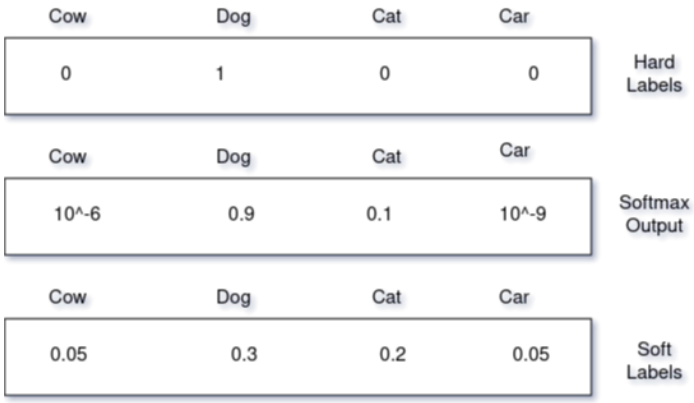

Figure 2.16: An example of hard targets and soft targets.

The introduction of softmax layers in the last layers of deep learning models reduces the probabilities of low probability classes while increasing the probabilities of higher probability classes. Because most of them are near zero, extracting relative information between classes is more difficult. To generate soft labels, a temperature (T) parameter is employed in the output softmax layer of the teacher model, as shown in Figure 2.1. The exact temperature is applied when training the student network. This parameter is set to 1 during inference time.

$$q_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \quad (2.1)$$

The authors of [14] created several experiences with other losses in the process of knowledge distillation. They found that the loss function described in 2.2 produced the best results.

$$L' = \sum_{i=1}^{n} \left\{ (1-\varepsilon) \Big[-\sum_{y=1}^{K} p(y|x_i) \log q_\theta(y|x_i) \Big] + \varepsilon \Big[-\sum_{y=1}^{K} u(y|x_i) \log q_\theta(y|x_i) \Big] \right\} \quad (2.2)$$

The loss function 2.2 comprises two cross-entropy losses, one for the soft labels target and the other for the dataset's hard targets. This loss function produced the best results and precisely adjusted the weight of each label via the alpha parameter.
This work uses this strategy to transfer information from a large model trained using the *Kinetics 400* dataset to a smaller model to fine-tune this student model to identify bullying on school grounds.

Chapter 3

State-of-the-Art

This chapter aims to present the previous studies made about bullying, seen through the eyes of psychological professionals and deep learning technologies used in tasks like action recognition on videos made by the scientific community.
The first section has the intention to show the studies already made by the scientific community, using the methods described in the second section, addressing bullying actions recognition. In this section, three articles are discussed with each one of their methods and results.

3.1 Bullying Recognition through Automated Video Analysis

3.1.1 Bullying Event Detection through Frame Sequence (Wang et al. [1])

The authors in [1] discussed the creation of a deep learning model that was able to detect with precise results violent actions performed by a group of people on school property. The model used the motion properties of video to predict a class label to a particular action. This was only possible with the implementation of two independent models that process pictures and videos. This ensemble learning achieved high accuracy rates because it focused on optical flow analysis, minimizing the background information and allowing precise movement analysis. The chosen architecture showed that it is preferential compared to 3D convolutions because these operations depend on high computation power and a higher train time.
This work used a collected dataset from different sources, including hockey videos, UCF-101 datasets, movies, and pictures downloaded from Google. The resultant dataset was composed of 193,155 fighting images and 224,665 non-fighting images. The total resources were based on 1649 videos with fighting actions, kicking and punching, and non-fighting such as playing soccer, talking, hugging, clapping, and waving.
The data has a significant role in developing deep learning applications, so the amount of data available representing a specific problem has a high impact on the system's performance. To achieve higher results, the authors in [1] used data augmentation to increase the number of samples in the dataset.
They used techniques such as Salt-pepper noise, Gaussian noise, Normal Blur, RGB value variation, Medium Blur, and RGB value being transferred to HSV value to the original images. This approach allowed the increasing of the dataset by a factor of 7, both in fighting and non-fighting frames.
Besides the spatial component of a video, the temporal element is also responsible for creating the animation of scenes. When analyzing the action recognition problem, some aspects must be considered, such as the necessity of extracting features from two or more subsequent frames. A neural network known as LSTM gives us the possibility of interconnecting a bunch of features collected from different structures.

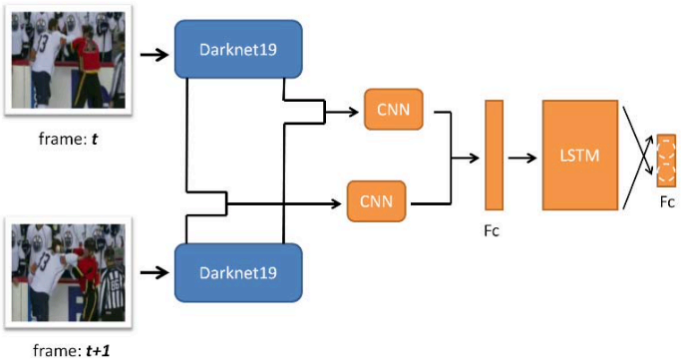

Figure 3.1: Model Architecture used for video-analysis process. Image from [1].

The work [1] aggregated CNNs with LSTM and optical flow algorithms to process a video. The resulting model is illustrated in diagram 3.1 and used two pre-trained Darknet19 CNNs, trained on the ImageNet dataset, each one receiving a frame from the video. The input of the CNNs was first processed by an optical flow algorithm to extract the motion features. The motion features were fed into two CNNs, following a fully connected layer and a LSTM. There were important aspects that made possible the excellent accuracy of the network. First, the use of pre-trained Darknet19 showed a performance increase; the use of optical flow allowed the extraction of motion between two frames, also increasing the efficiency. Finally, the LSTM network, capable of processing a data sequence, allowed extracted features to take into account two frames.

Together with the model 3.1, another model was developed with the purpose of image classification. The model was implemented using a CNN and a transfer learning process. From all the used CNNs, GoogLeNet, AlexNet, and ResNet, ResNet achieved higher accuracy. The ResNet model has some construction properties that make it a good classification model. The residual block represented in the Figure 3.2 allows the network to propagate features from early layers, with high semantic information, into layers with low semantic information. Besides that, it also solves the problem of vanishing gradient.

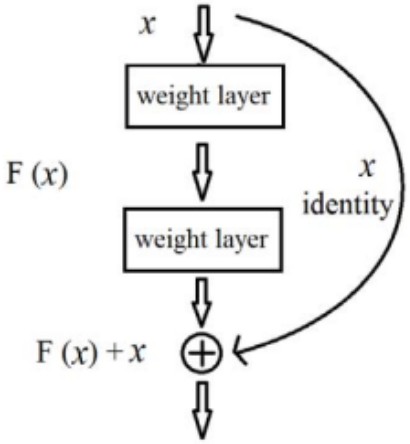

Figure 3.2: Illustration of the residual block on ResNet architecture. Image from [1].

Most of the bullying events have similarities with sports actions or hugging movements. Us-

ing a model that classifies images in violent or no violent actions improves the system in recognition of bullying scenarios. So, the resulting system presented in the diagram 3.3 shows the input data that are treated apart into two models, frame-image classification and frame-sequence classification. This fusion leads to an ensemble learning method resulting in one output result.

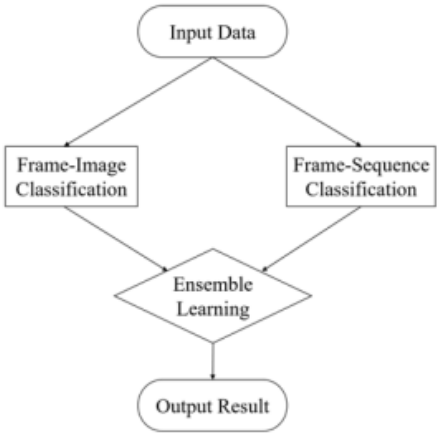

Figure 3.3: End model containing the ensemble learning method. Image from [1].

One of the factors needed to take into account for training the ensemble model was the number of consequent frames used in the frame-sequence classification model. The model was tested with 6,8,10,12 and 14 resulting frames, and the results are shown in table 3.1. All the results displayed at [1] were achieved using a GPU NVIDIA GTX1080.

Threshold (frames)	Accuracy
6	86.36%
8	86.36%
10	90.90%
12	90.90%
14	81.81%

Table 3.1: Results achieved with different number of consequently frames. Image from [1].

Table 3.1 shows that 10 or 12 consequent frames achieve the best results for predicting bullying actions in the frame-sequence classification model.
The frame-image classification model also was tested and was created one table, represented in 3.2, comparing the three different architectures used in the training phase.

Method	Accuracy
AlexNet	97.01%
GoogLeNet	98.17%
ResNet	99.33%

Table 3.2: Accuracy values achieve in test time using different CNNs architectures. Image from [1].

The results presented in Table 3.2 shows ResNet architecture had the highest accuracy with 99 percent in the test dataset. In conclusion, the use of an ensemble learning method with one model for frame-sequence analysis and another for image classification allowed the authors at [1] to build a system capable of detecting bullying actions, processing 75 fps, and having

accuracy close to 91 percent. This was only possible with the help of computer vision and deep learning technologies like CNNs, optical flow algorithms, and LSTMs. This work has a high practical value, changing the dynamics of seeing rude behavior from young students.

3.1.2 Campus Violence Detection Based on Artificial Intelligence (Ye et al. [2])

The authors at the following work [2] strove to implement a system able to detect campus violence through surveillance cameras with the capture of sound. They studied the subject and found out that most previous articles used sensors, video analysis, and emotion classification through sound to predict violent actions. Most of the investigations only focused on social events, like fighting or using weapons to intimidate a person, but none studied campus violence. The main difference between these two types of events is usually the nonexistence of resistance from the victim and no use of weapons from the aggressors. One of the things that makes the recognition of campus violence more challenging to identify than social violence is the presence of intimidation and threats.

After the study was done, the authors at [2] needed to contact some volunteers to replicate some violence at campus and record them, intending to build a dataset describing the problem. In the next stage was the need to create a pre-processing of the data to transform the video in frames, achieving 12,448 frames of campus violence and 12,448 frames of daily-life activities. It was suggested that 16 frames in one processing unit result in a tensor of dimensions 3x16x112x112, as shown in Figure 3.4. They normalized the size of the images into 112x112 because it was a size that balanced recognition accuracy and real-time performance.

For the speech emotion recognition, the authors used two private databases, Finnish emotional and Chinese emotional. The Finnish database contains content about explicit school bullying, which had an important role in their work, both of them with either pure emotions or mixed feelings. The third was the CASIA public database, divided into three types of pure feelings: angry, sad, and happy. The aggregation process of the three datasets resulted in a ratio of positive and negative samples by 1:1.

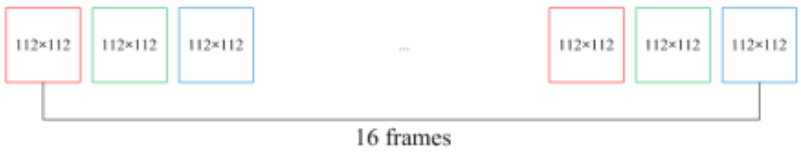

Figure 3.4: Processing unit with 16 frames, each one with 3 color channels and dimensions 112x112. Image from [2].

Although some studies showed different deep learning approaches capable of feature extraction from a video, the authors chosen a 3D Convolution Network to feature extraction and a 4-layer neural network for action classification.

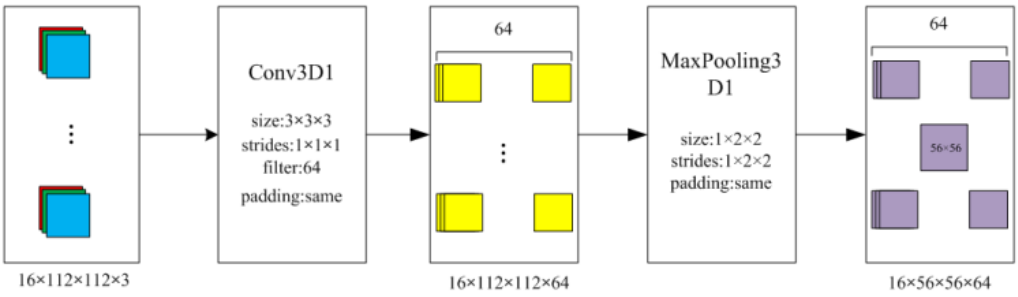

Figure 3.5: 3D Convolution network architecture with the purpose of feature extraction. Image from [2].

The Figure 3.5 shows that the first architecture chosen by the authors has eight 3D convolutions and four 3D maximum pooling operations, with different filters and input dimensions. The first one has an input layer with 16x112x112x3 dimensions; the next layer is a 3D Convolution with 3x3x3 kernel dimensions and 64 filters. The convolutions result in a tensor with 16x112x112x64 dimensions fed to the 3D Polling layer reducing the space tensor for 16x54x54x64. The second model implemented by the authors returned a classification taking into account the feature vector given by the feature extraction network.

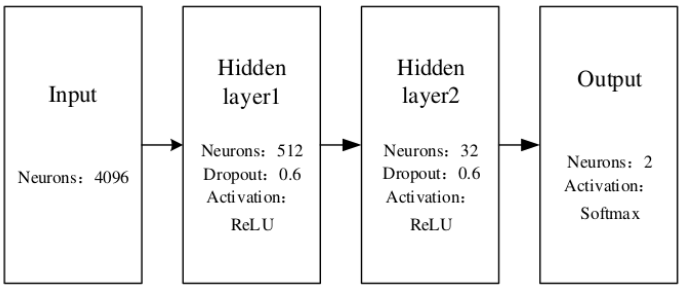

Figure 3.6: 4-Layer network with 2 hidden layers and output layer predicting 2 classes. Image from [2].

The Figure represented in 3.6 shows that the model input is a 4096-dimensional feature vector extracted by the 3D Convolution network described in 3.5. There are two hidden layers with dropout 0.6 and ReLU activation functions. The network's end comprises two neurons and a softmax operation, returning a two-class classification.

Most of the bullying events occur with violent actions and verbal expressions where the victim shows nervous and frustrating feelings in opposition to the aggressor that shows angry ones. A fine line expresses the difference between social violence and bullying. The use of emotion recognition can significantly improve the performance of the system. That way, the authors implemented a neural network for speech emotion recognition using MFCC speech feature extraction for speech recognition.

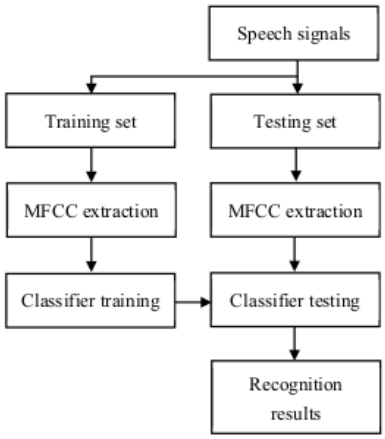

Figure 3.7: Speech emotion recognition pipeline with MFCC feature extraction method and classifier. Image from [2].

Figure 3.7 shows the pipeline implemented to extract the features from the speech signals and use them for training the classifier allowing the emotion recognition process.

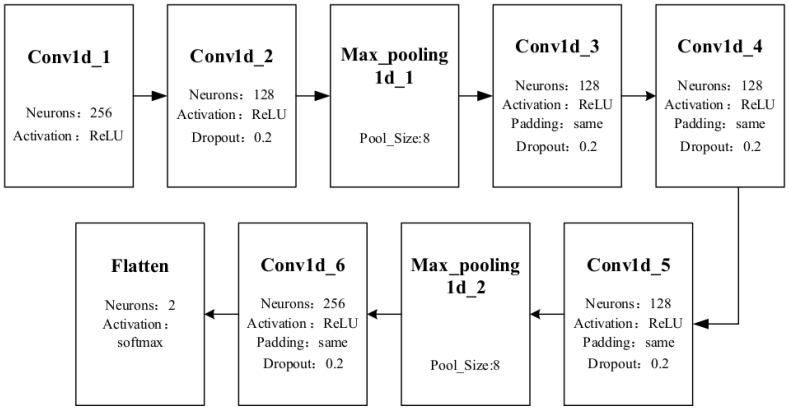

Figure 3.8: Neural network architecture used for bullying emotion recognition. Image from [2].

After extracting the features from the speech signals, the vectors are fed into a neural network classifier presented in Figure 3.8 with six 1D convolution layers, two max-polling layers, and ReLU activation layers. The use of dropout avoids over-fitting and increases the generalization of the model.

3.1.3 A Skeleton-based Method for Recognizing the Campus Violence (Xing et al. [3])

The authors in [3] used human pose estimation to gather information about different types of actions and interactions between two persons or a group. The main focus of the work was to create an application able to recognize bullying and non-bullying events on campus. They studied a large variety of methods to recognize actions in a video and realized that some of them were not satisfactory because the background features were complex. The brightness of different types of lights influenced the extraction of features.

They proposed a new method to recognize actions in a complex background through human pose estimation methods. The method used to extract information about the action and discard the background information was using an ST-GCN and 2S-AGCN with an attention module to form a 2s-AAGCN, making possible the classification of actions through a spatial and temporal dimension. Each node of the graph corresponds to a joint and the edges to bones of the human body.

The dataset used in the experiments was NTU-RGB+D with 56,880 action samples in four different modalities like RGB videos, 3D skeletal data, map sequences, and infrared videos.

The authors relied on existing dataset NTU-RGB+D containing 56,880 actions clips corresponding to 60 action classes to train and test the network. The dataset's creation involved three Kinect V2 cameras and has four modalities. Only two of them were relevant to the study, RGB videos 1920x1080 and 3D skeletal data expressing 25 body joints at each frame. Most of the action classes are in a single person, which is inconsistent when considering the violence recognition problem. The authors only selected the actions clips that represented two-person interaction like, punching, kicking, and pushing.

For the work of bullying events detection addressed at [3] the authors transformed the 2D skeleton information returned by the OpenPose method into 3D data. The OpenPose method is a SOTA method for estimating human pose using 18 joints as depicted in Figure 3.9.

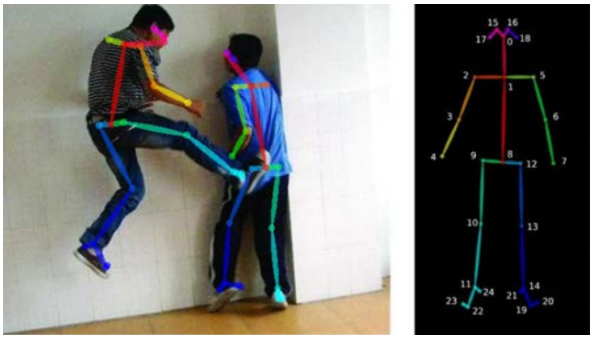

Figure 3.9: OpenPose applied to a violence event in a school on the right. On the left the label with respect of each joint point. Image from [3].

The GCNs are receiving much attention in the area, allowing the development of a new variety of applications. The human estimation pose problem is one of the applications of the GCN because the pose of the human body can be expressed in a better way through a graph than a linear vector or a grid. The graph can maintain the body's structure when using other data structures loses information about length and interactions between joints of the body.

The authors in [3] used the spatial configuration partitioning strategy created by the authors in [16] on the model ST-GCN. This technique is helpful for the construction of the graph and describes how the joints are labeled. They chose the model 2S-AGCN [17] as their base model with additional attention components on each block of the network.

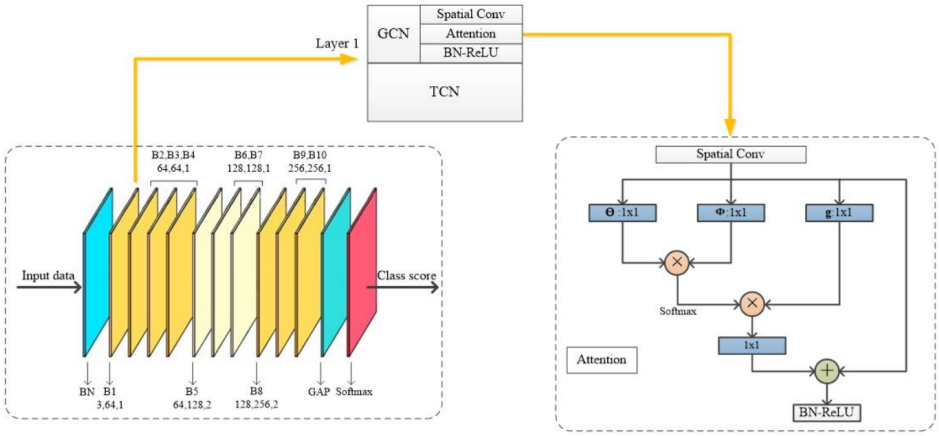

Figure 3.10: Additional attention module incorporated in each block of the AGCN network, creating a new AAGCN block. Image from [3].

Figure 3.10 shows that the network is composed of ten basic blocks; in each block was added an attention module between the spatial convolution and the batch normalization layer with ReLU.

The attention mechanism illustrated on the right shows three convolutions of 1x1 kernel dimensions applied on the features returned by the spatial convolution, next perform a matrix point multiplication operation to achieve the features auto-correlation and at the end a Softmax operation to obtain the attention coefficient. The last operation resides on a residual connection with the original input feature map.

The ST-GCN model only gives information about the joints but does not take into account the direction or length of the bone. To do this, the AGCN model was implemented with two streams, J-AGCN to estimate the joints information, and B-AGCN to retrieve bone information, as presented in the Figure 3.11.

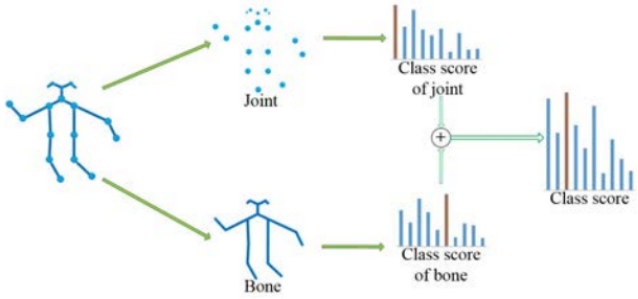

Figure 3.11: Illustration of the two-stream process to achieve information about the joints and bone. Fuse information to predict a class label about the action. Image from [3].

The final score is achieved through the fusion of the information of each stream, concerning bones and joints information.

In the experiments carried out in [3], the results were achieved in Ubuntu 18.04 operating system and with Nvidia GTX 1080Ti. The NTU RGB+D dataset was reconstructed to contain only classes concerning the bullying problem and was used to train and test the networks.

The X-subject setting was used to calculate the model metrics and was applied by selecting training clips with one subset of actors and testing clips with the remaining ones.

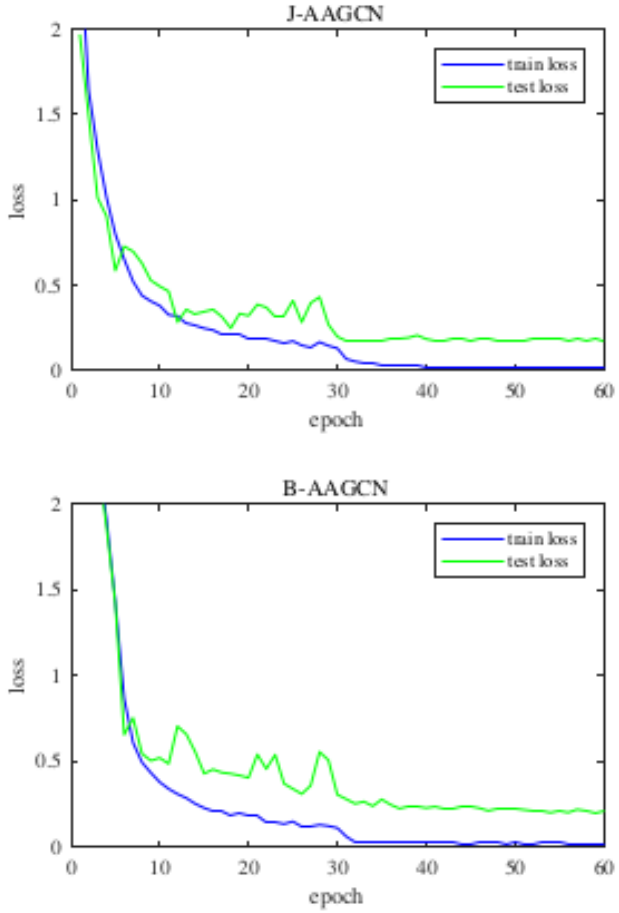

Figure 3.12: Loss curves of method 2S-AAGCN. Image from [3].

Methods	Accuracy(%)
ST-GCN [15]	81.54
J-AGCN	93.75
B-AGCN	92.77
2s-AGCN [20]	94.76
Our method	**Accuracy(%)**
J-AAGCN	94.37
B-AAGCN	93.64
2s-AAGCN	96.07

Table 3.3: Accuracy achieved using different methods to action recognition through GCN. Image from [3].

Through the analysis of Figure 3.12, it is possible to observe that the process of training and testing of the 2S-AAGCN model achieved good results with a decreasing loss curve in both phases. It is plausible to recognize that the 2S-AAGCN model achieved a better performance in the bullying action recognition using the NTU-RGB+D dataset. Table 3.3 shows the comparison results of the accuracy achieved in different methods.

Chapter 4

Datasets

This chapter describes the two datasets used in the development phases, training and evaluation, of each deep learning pipeline. The first dataset was used for fine-tuning and knowledge distillation purposes. It is the most common set used in the research community as a baseline for performance competitions. The second is the dataset proposed in this thesis, representing young students in a school environment in bullying and non-bullying actions.

4.1 Kinetics 400

The *Kinetics 400* dataset was produced in response to a scarcity of large datasets for various video analysis applications. It became a well-known dataset for performance benchmarks due to the ability to build a deep learning model from scratch and the demanding training and assessment processes in multiple model architectures. The good results of fine-tuning with pre-trained models on the ImageNet dataset for image classification tasks inspired the authors to create a large and encompassing dataset for video analysis. The models trained with the Kinetics 400 dataset have become a suitable feature extractor to improve the performance of training deep learning models for specific tasks with reduced datasets through the fine-tuning process.
This dataset contains 306,245 films divided into 400 different human activity classes, each with at least 400 videos. Each movie has a chronology of around 10 seconds, and there are no untrimmed recordings. YouTube served as the primary source for these videos, with *Amazon Mechanic Turk* serving as the labeling interface. The dataset is a collection of CSV files corresponding to different stages of model development, with each entry including the YouTube URL, the start and finish chronology of the human influence, and the label. These files are read by a Python script, which download and trim the movies to a local folder.

Data Gathering For the Kinetics 400 dataset data collection, one script read the three CSV files downloaded from the official website to lower the dataset's dimension by 10% and store the same number of videos for each class. Following the execution of the script, three CSV files were saved in a single destination folder containing only the annotations for downloading 30,624 films.
The next step was to rewrite the script for downloading and cutting the movies from the repository [18] because the multi-threading procedure prevented the films from being correctly saved in the folders. To do this, a new script was written to download the movies one by one, utilizing a dictionary to hold the information from the CSVs and the *youtube-dl* command-line tool to download them.

The movies were kept in a single folder with 400 folders with respect of each label. Some videos were not downloaded throughout this procedure due to protected rights and non-existence. The download and pruning of the 30,624 movies took around 12 days using this method.

The final step was writing a new script to read the three CSV files into a dictionary and trimming the films with the start and end timeline annotations using the *ffmpeg* tool.

Data Pre-Processing The *Kinetics 400* data that was downloaded will be used in the future chapter to construct the knowledge distillation procedure. The knowledge distillation model contains two paths, one for RGB frames and one for optical flow. This element entailed writing a script to read all of the movies and compute the optical flow for each frame before appending it to the frame. This approach produced a single array with five channels instead of three for each frame, considering the x and y axes of each optical flow. Each video was then saved as a *numpy* compressed file.

Including optical flow from the model meant a significant increase in computer resources, particularly memory, which was one of the primary reasons the complete kinetics-400 dataset was not employed in the knowledge distillation process. After all, even with compressed files, the memory usage for each film was 2x or 3x more than usual, easily exceeding the 10 TB storage limit.

After storing the necessary *numpy* files, a script was written to search the names of each file in the folders and generate a list of tuples, including the absolute path and the hard target label in numeric format. Having the information in a single list of tuples, it was shuffled and divided, with 70% used for training, 20% for validation, and the remaining 10% for testing. The data was stored in three CSV files, one for each development step.

4.2 You Never Forget

The *You Never Forget* dataset was built as part of this dissertation and attempts to establish a unique bullying dataset to identify bullying in schools. The information gathered in Section 2.1.1 demonstrated the necessity for constructing a new bullying dataset that included not just violent actions between students but also making fun of victims, hurling items at them, and stealing objects.

This dataset was created using recorded footage of four kids in various school settings, including classes, bathrooms, corridors, comfort zones, and outside. It features 550 movies, 350 of which were shot using two cameras, one static on the wall and a smartphone camera that provided a new perspective on the subject. The remaining 200 films were obtained from the WorldStarHipHop [19] website and were captured with smartphones and surveillance cameras containing young person behavior on school grounds. The recorded videos have a resolution of 1280x720 and a frame rate of 30 frames per second. Each downloaded video has a distinct resolution and 30 frames per second.

Figure 4.1: Frames that show the dataset recorded with students in school property. The first three images on the top show normal actions in halls, and classrooms without bullying. The last three images show bullying actions where a a group of students steal a bag and make fun of the victim, and throw objects to the victim that is writing.

Data Labeling Adobe Premiere software was used to load and clip the student's non-bullying, and bullying acts in trimmed films for labeling the recorded and downloaded footage. Because the program does not support the Avi codec, the movies were converted to a different format before loading. At the same time as the format changed, the existing audio in each video was deleted using the *ffmpeg* program. The annotated films were kept in one folder, divided into two files with respect of each class.

Figure 4.1 exhibits video frames collected on school grounds of four students participating in bullying and non-bullying activities. The first three pictures show non-bullying behavior in classrooms and rest places. Bullying, violent conduct, stealing a school bag and a ball, mocking victims, and throwing objects are all represented in three-down illustrations.

Data Pre-Prossessing The resolutions of the downloaded videos varied. Some had black borders; when resized, essential information was lost due to the smaller frame size. A script was written to count the number of pixels in each frame video's borders and clip them to increase the frame size.

Following the data labeling and cutting steps, a script was created to detect existing text in each frame of the downloaded movies. The script detected the *OCR* characters and returned the position on the frame using a pre-trained model from *keras*. Upon the determination of the location, a procedure was developed to paint the picture pixels the same color as their neighbors. Unfortunately, because some of these texts are not in the same places throughout the film, the removal procedure caused issues when computing the optical flow. As result, this approach was not employed during the development phases.

The same optical flow technique used in the *Kinetics 400* was then utilized to compute the optical flow and save the data in compressed *Numpy* files. After storing the data, the absolute pathways and hard target labels were saved on a list of tuples, and a split was performed to construct the train, validation, and test segments. The numpy files were kept in a single folder comprising three folders for each development phase and two folders for each label.

The absolute routes and hard targets were saved in three CVS files that will be loaded during the following development session.

Chapter 5

Proposed Deep Learning Pipelines

This chapter shows the programming details of the proposed deep learning pipelines for recognizing bullying activities in videos. The first section is separated into five steps, each corresponding to a particular model implementation and its dataloader, learning rates, frame rate, and techniques for training, validating, and testing the model. The last step presents the implementations of the *Knowledge Distillation* technique, used to train the *FGN* with the *Kinetics 400* dataset and further fine-tune it with the *YNF* dataset.

5.1 Flow Gated Network

The *Flow Gated Network* was discussed in Chapter 2 and was downloaded from [20]. The authors in [13] developed a unique real-world fight video dataset captured by security cameras, and the model was deployed to recognize fight events in real-world scenarios. Another feature of this model is its small size (272,690 parameters), which allows it to be deployed on IoT devices with limited computing power. This is why it is employed in this dissertation, and it will be developed to identify bullying in real-world circumstances.
In the development phase it was used the *PopOs 20.04* operating system, *Pycharm Community* IDE, with *CUDA 10.1*, *Cudnn 8.0.4* and *Python 3.6.13*. The main frameworks used was *Tensorflow-gpu 1.13.1*, *Keras 2.3.1*, *Numpy 1.17.0*, *Matplotlib 3.3.4*, and *Scikit-learn 0.20.0*. For processing it was used a *Nvidia RTX 2070*, 16 GB's of *RAM*, *Ryzen 5* processor and 500 GB's of SSD storage. To train the model from scratch, it is necessary to modify the path variables in the file `RWF/Networks/Flow_Gated_Network.py` and execute it with a python environment.

Dataloader The dataloader was implemented through a *keras* custom class *Sequence* inheritance allowing the customization of the data generation, storage, and loading batch processes. This class receives from input the dataset directory path containing the numpy's files created in Chapter 4 related to the *You Never Forget* dataset, the batch size, and logical variables for shuffling and data augmentation operations.
For each numpy file, the dataset object instantiation scans the folders in the directory file and constructs a path and one hot label list. Each iteration of the load batch procedure includes an index list for iteration and a data generating function to load the batch data from numpy file. The frames are sampled in the load function to provide a temporal dimension of 64 frames for each video. When the video size is less than 64 frames, it repeats the video frames to attain the fixed value of 64; when the video size exceeds 64 frames, it generates another batch for the following 64 frames.

The default sampling function only generated one temporal batch, computing a dynamic sampling rate for each video frame portion with the fixed 64 temporal dimensions. To sample fixed rates for performance measurement, a new sampling rate function was designed. Following the loading and sampling operations, data augmentation via color and flip frame adjustments is feasible.

Learning Rate The learning rate has a significant influence on the training phase of any deep learning pipeline, and it is determined by the model and dataset characteristics. To find the best solution for the *You Never Forget* dataset trained from scratch, several learning rate settings were employed, as well as fine-tuning and K-Fold Cross Validation. The following learning rates were used: 0.05, 0.01, 0.005, 0.001, 0.0005, and 0.0001.

Sampling Rate The sample rate is a vital hyper-parameter that should be adjusted to get the best solution. The general value begins at two and increases in powers of two until it reaches the value of eight. Four alternative experiments were created during the development of this model for the following sample rates: 2, 4, 6, and 8.

Fine-Tuning Fine-tuning is the process of retraining pre-trained models to a given purpose utilizing large datasets such as *Imagenet* or *Kinetics 400*. This procedure will be used with the knowledge distillation technique to fine-tune the model that was previously trained on a subset of the *Kinetics 400* dataset and retrain it to identify the bullying assignment addressed in this dissertation. The fine-tuning technique emerged from the excellent results obtained when using pre-trained models to build deep learning models for a given task with little training data.
To implement this strategy, a script was written that would load a *Keras* model, delete the network's final fully connected layers, and replace them with new ones, adjusting the number of neurons to the number of classes addressed in the new task. The other layers were frozen after replacing the fully connected, and only the new ones were trained.
After training the last fully-connected layers, the frozen layers were unfrozen, and a new training process was started, with few epochs, to make just little adjustments to the weights of the model.
For running this process, it should change the path variables in the file `\detokenize{RWF/Networks/TransferLearning_Finetuning.py}` and execute it with a python environment.

K-Fold Cross Validation K-Fold Cross Validation is another way of evaluating deep learning applications. This procedure includes training various models with distinct sections of the train, validate, and test datasets. Finally, the final forecast is the average of the models. This approach is often applied in competitions because it improves model accuracy and allows the training of models with fewer data. Each model is tested with different data providing them with new information.
This method is used in the model *FGN* to train an ensemble model from scratch using a much-rich testing dataset. To do this, a script was created using a new dataloader that divided

the dataset into 10 test splits through a function from the *skicit-learn* package. Each one of these splits originated a training, validation and a test dataset. The *Keras API* was then used to define the model, and each model, with the exactly same architecture, was trained in cycle. For executing this process it is necessary to modify the paths variables in the file `RWF/Networks/Flow_Gated_Network_K.py`, and next execute it with a python environment.

5.2 C2D

The C2D model was one of the first deep learning models to analyze and classify videos. It was mentioned in Chapter 2, where the model operations and layers were explained. This model implementation was downloaded from the repository [21], which contains configuration files for various models. This dissertation uses this model for comparative reasons with deep learning SOTA algorithms.
In the development phase it was used the *PopOs 20.04* operating system, *Pycharm Community* IDE, with *CUDA 11.1*, *Cudnn 8.0.4* and *Python 3.8.11*. The main frameworks used was *Torch 1.9.0*, *Torchvision 0.10.1*, *Tensorflow-GPU 2.8.0*, *Numpy 1.22.3*, *Matplotlib 3.4.2*, *Scikit-learn 1.0.2*, and *Detectron2 0.5*. The installation process is described in [21]. For processing it was used a *Nvidia RTX 2070*, 16 GB's of *RAM*, *Ryzen 5* processor and 500 GB's of SSD storage.

Configuration File The main script is stored in the *tools* folder, and to run the training process *run_net.py* script should be executed with the input `--cfg` related with the path to the configuration file. The configuration file to the C2D model is stored in the folder `configs/YNF` with the name *C2D_8x8_R50_IN1K.yaml* of this dissertation repository [1].
The configuration files are critical for creating complex deep learning pipelines because it is possible to change various variables and hyper-parameters without modifying the code. In this configuration file is possible to control the development phase, training, testing, or inference, change the learning rate, path to the datasets CSVs files, number of epochs, batch size, log paths, visualization tools, and model characteristics.

Dataloader The dataloader for this model is already present in the main repository [21], under the folder `SlowFast/slowfast/datasets` with the name *kinetics.py*. This dataloader is built as a custom class extension of the *Pytorch* class *torch.utils.data.Dataset*, changing the inputs of the class constructor and the method `getitem__` overridden for loading and constructing the kinetics dataset batches. This file also includes data augmentation routines that modify each video frame's color, rotation, and crop settings. This dataloader loads the *You Never Forget* dataset after creating the CVS files described in Chapter 4 with the same annotations as the *Kinetics 400* dataset.

Learning Rate The learning rate is a hyper-parameter that regulates the changes in each step of the update for each weight, and its selection is critical to achieving the best solu-

[1] `https://github.com/EdgarDaniel/MSc_PreventingBullying`

tion. The creators of [21] designed a dynamic learning rate that begins with larger values and decreases with each epoch. This strategy was not applied in this dissertation; instead, a predefined learning rate was used. To accomplish this, a modification was made to the path-SlowFast/tools/train net.py code to keep the learning rate constant throughout the train. The experiments were created with the following learning rates in mind: 0.05, 0.01, 0.005, 0.001, 0.0005, and 0.0001.

Sampling Rate Another hyper-parameter is the sampling rate, which searches for relevant frames to describe the problem. For example, in films, subsequent frames nearly always include the same information as the preceding one. Therefore the main goal is to find the ideal number that discards comparable frames and only selects those with a significant influence on achieving the best solution.
The sampling rate number in this model implementation will be constant and vary between 2 and 8, double the previous one. Each experiment will be done at the following frame rates: 2, 4, 6, and 8.

Fine-Tuning This model's fine-tuning procedure differs from that described in the prior model. The model zoo of the repository [21] already contains a pre-trained *Kinetics 400* model. However, this file was corrupted when downloaded. Therefore, a search was made to find another pre-trained C2D model file. This search occurred on the issues area of the repository. During the search, the same problem appeared to another person, and a new weights file was found and shared with the public. It is available for download in the issue located on the web page [22].
It was essential to change the configuration file after downloading the model, adding a new option named `checkpoint_file_path` with the path to the downloaded pre-trained model.

5.3 I3D

The Inflated 3D model was built to improve the well-known 2D convolutional neural networks for image classification, adding another channel to reflect the time dimension for training these well-designed models with movies. This model was also described in Chapter 2, and it is implemented in this dissertation to compare performance to the SOTA methods.
The inflated model used in this implementation was a *Resnet 50* model, which can be found in the same repository as the C2D model discussed above. This implementation uses the same tools and frameworks as the previous one.

Configuration File The configuration file for running the train and test phases for this model is named `I3D_8x8_R50.yaml`. It can be found in the repository addressed to this dissertation in the folder `SlowFast/configs/YNF`. The data path, the batch size for training and testing, the number of epochs, metrics, evaluation epochs, and log settings have all been changed.

Dataloader The identical *kinetics.py* file and CSV files mentioned in the chapter 4 were used to load the data in this model.

Learning Rate The learning rate used for this model was constant with the following values: 0.05, 0.01, 0.005, 0.001, 0.0005, and 0.0001 .

Sampling Rate The sampling rate used in this model's training and test phases was also fixed along with the processes with the following values: 2, 4, 6, and 8.

Fine-Tuning The fine-tuning procedure for this model required downloading the pre-trained model from the model zoo repository web page [21], which had the same name as the configuration file `I3D_8x8_R50.yaml`. The configuration file was then modified to replace the path of the downloaded model in the field checkpoint file location.

5.4 SlowFast

One of the SOTA deep learning algorithms for categorizing and recognizing actions in movies is the *SlowFast* model. It employs two paths, each with an inflated Resnet50, to learn the slow and rapid information in the videos, as explained in Chapter 2. The *Facebook Company* introduced it and was chosen for this dissertation because it looks the most promising.
The *SlowFast* model is implemented in the repository [21], which was already mentioned in the previous two models, what implied the use of the same *Python* environment and tools.

Configuration File The configuration file required in this model was previously implemented in the repository [21], as indicated in the previous two models. Changes were made to adapt to the *YNF* dataset, such as the number of epochs, batch size, learning rate, dataset pathways, and log settings.
The final configuration file, `SLOWFAST_8x8_R50.yaml`, may be found in the repository citation devoted to this dissertation under the folder `SLOWFAST/confis/YNF`.

Dataloader The same dataloader implemented in the previous two models was used to load the data to train and test the *SlowFast* model.

Learning Rate Were made several experiments related to the learning rate with the following values: 0.05, 0.01, 0.005, 0.001, 0.0005, and 0.0001.

Sampling Rate The results are shown in Table 6.3 in Chapter 6 and include test metrics for the following values: 2, 4, 6, and 8.

Fine-Tuning The pre-trained model was acquired from the model zoo website [21], which has the same name as the *SlowFast* configuration file.

5.5 Knowledge Distillation

Knowledge distillation is a compression approach that allows information to be transferred from a teacher or ensemble model to a student model with fewer parameters and without requiring extensive computing resources. The fundamental goal of this method is to develop a model that can be run on low-power devices while maintaining the same accuracy as the teacher model. Because the inference is made on the device, sending information to high-performance servers in the cloud is unnecessary, which reduces model deployment costs and increases security.

The main reasons for implementing this method in this dissertation on the *Flow Gated Model* are the lack of a pre-trained *Kinetics 400* to compare the results obtained through the fine-tuning process of other models. The need to reduce deployment costs for schools and data security guarantees, allowing it to run entirely on surveillance cameras.

The implementation of this model is divided into three steps. The first is to download 10% of the *Kinetics 400* dataset and use it to compute the *SlowFast* pretrained instructor model logits. The second part involves establishing a new dataloader and using the knowledge distillation method to train the *Flow Gated Model*. A new model with a customized loss function was created to do this. Following training, the best model was kept and applied to fine-tune with the YNF dataset in the third stage. The space and time required to download and store the *Kinetics 400* dataset posed various challenges during the development of the first two stages. Two weeks were necessary to download and trim 10% of the dataset, 600 GBs to store the optical and video information, and approximately 25 days to train just one experiment with 150 epochs.

To develop and execute this model, it was used a *Geforce GTX 1080*, with 32 GB of RAM, 128 GB's for primary storage and 2 TB to secondary storage and a Xeon E3-1200 v6/7th Gen Core Processor. The main tools and frameworks was the *Pycharm Community IDE*, *Python Interpreter 3.8.10*, with *CUDA 11.2*, *Cudnn 8.1.1*, *Tensorflow-gpu 2.8.0*, *Keras 2.8.0*, *Numpy 1.22.4*, *Scikit-Learn 1.1.1*, *OpenCV 4.5.5.64*.

Dataloader A *Pyhton* script was implemented using the *Sequence* inheritance class from the *Keras API* to generate custom functions to load each batch to create the dataloader for the *FGN* trained with the *Knowledge Distillation* approach. This procedure included two distinct steps. The first was to run the *SlowFast* model inference on 10% of the downloaded *Kinetics 400* dataset to save the predictions, hard targets, and paths to each video in three numpys files, one for each development step. The second stage involved converting each video to numpy compressed files comprising the video frames with the corresponding optical flow. The custom dataloader function reads the numpy file created with the *SlowFast* model and produces the appropriate data. The custom dataloader function reads the numpy file generated by the *SlowFast* model, creates the appropriate paths to the numpy compressed files, loads the data, and reconstructs the batch size using the hard labels and teacher predictions.

Model Implementation The model implementation was made through the custom model functions in the *Keras API* due to the need for creating a custom loss function presented in Chapter 2.

Learning Rate Different experiments were prepared for the learning rate hyper-parameter in order to find the optimal value. The results for the single event with a learning rate of 0.001 were not completed due to the time required to train the model.

Sampling Rate For the sampling rate hyper-parameter only the value of 4 was used. None result was determined because the training process was not finished.

Chapter 6

Results / Discussion

This chapter presents the results obtained during the testing phase of each deep learning pipeline to find the best model, hyper-parameters, and approaches for identifying bullying in schools. The first section demonstrates the outcomes of training deep learning models from scratch. The second section contains a table showing the results obtained by deploying an ensemble model and training the model using the K-Fold Cross Validation approach. The final section describes approaches for fine-tuning the C2D, I3D, and SlowFast models that have already been pre-trained on the *Kinetics 400* dataset.

6.1 Training from Scratch

The deep learning model *Flow Gated Network* mentioned in Chapter 2 was trained from scratch with the *YNF* dataset. For this, different experiments were made to search for the best hyper parameters for the optimal solution. From scratch, the only model capable of converging to a minimum was the *Flow Gated Network*. The training and validation results can be seen in Figure 6.1

The left plot in Figure 6.1 shows the loss metric along with the training and validating process with the *Flow Gated Model*, trained from scratch on the YNF dataset. The train and validation losses decrease along the process, achieving their lower values in the 15th epoch. The training process was made for 50 epochs, but after the 25th epoch, the validation loss raised, showing the beginning of overfitting.

The right plot in Figure 6.1 shows the accuracy metric for the training and validating process with the *Flow Gated Network*. It can be observed that it achieved the highest validation accuracy value of approximately 70% in the 25th epoch.

B	0.78	0.22
NB	0.37	0.63
	B	NB

Figure 6.2: FGN Confusion Matrix

Figure 6.2 shows the percentage of the true positive and false positive values achieved on the *YNF* dataset in test with a threshold of 0.5. It can be concluded that the model classifies the bullying actions with 78% of accuracy, and the non bullying actions with 63% of accuracy.

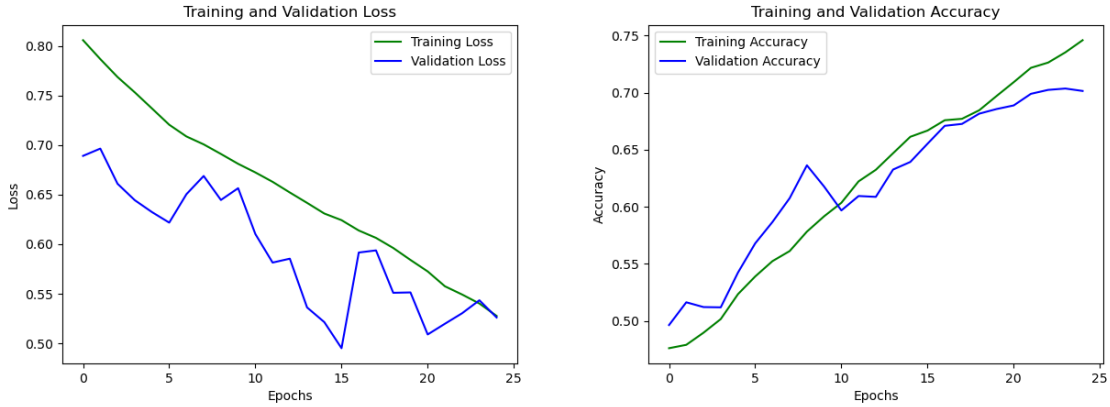

Figure 6.1: Loss and Accuracy plots related with the training and validating processes with the *Flow Gated Network*, trained from scratch on the YNF dataset. These results were achieved through a process of 25 epochs, 0.0001 for learning rate, no data augmentation, with the optimizer SGD, batch size of 2, the frame rate of four, and weight decay on the last two dense layers.

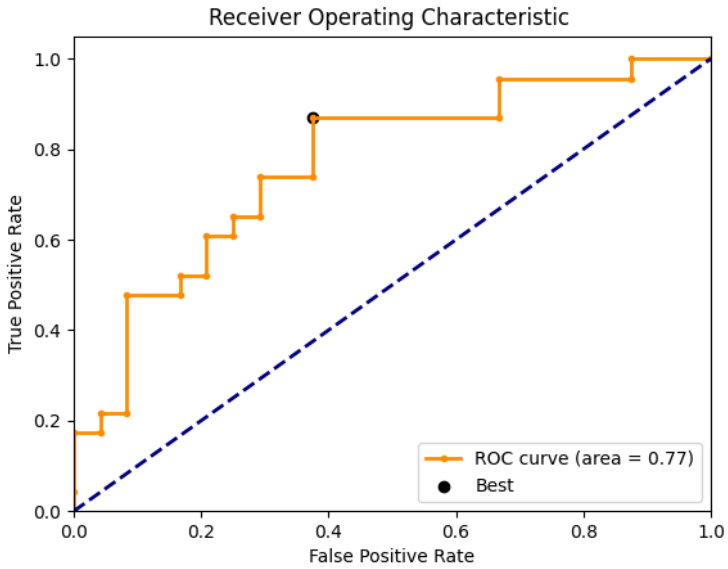

Figure 6.3: ROC Curve for the bullying class achieved with the *FGN* model. The best test threshold can be seen on the black dot.

Figure 6.3 shows the *Receiver Operating Characteristic* curve respective area with value 0.77. The optimal threshold value was calculated and can be seen in Table 6.4. These metrics show that the model was capable to achieve good performance.

Other experiments were made in the same conditions for different sampling rates to search for the suitable value for the *YNF* dataset. The results can be seen in Table 6.1.

YNF Dataset	
Sampling Rate	FGN (%)
2	57.45
4	70,21
6	48.93
8	57.45

Table 6.1: Top1-Accuracy for models trained from scratch on the YNF dataset on test dataset with different sampling rates. The accuracy was calculated with a threshold of 0.5 .

Table 6.1 shows the accuracy percentage on the test dataset, for different sampling rates. The highest accuracy occurs with a sampling rate of 4.

6.2 K-Fold Cross Validation

This section addresses the K-Fold Cross Validation technique used when the dataset has few instances. The main idea is to split the entire dataset into of test sets. The objective is to train different model instances with the exactly same architecture, each with a different test portion, making it possible to test the entire dataset. The final result is the average of the accuracy achieved in each model. This method usually raises the accuracy of the ensemble model due to the knowledge spread to the K models. The test accuracy for each model can be seen in Table 6.2.

YNF Dataset - K/Fold	
Model Number	FGN (%)
1	57.14
2	80.36
3	62.50
4	56.36
5	70.37
6	68.52
7	57.41
8	64.82
9	72.22
10	62.96
Average	65.67

Table 6.2: Top1-Accuracy achieved in test dataset using the K-Fold Cross Validation technique. These results were achieved through a process of 50 epochs, 0.0001 for learning rate, no data augmentation, with the optimizer SGD, batch size of 2, the frame rate of four.

Table 6.2 shows the accuracy achieved on each test dataset portion created through the method K-Fold Cross Validation. The ensemble model has 65.67% of accuracy when averaging the results.

6.3 Fine-tuning

The *SlowFast*, *I3D*, and *C2D* models only achieved suitable results when trained with the weights of the *Kinetics 400* dataset. This is because these models need more data to train from scratch, and the *YNF* dataset only has 550 videos.

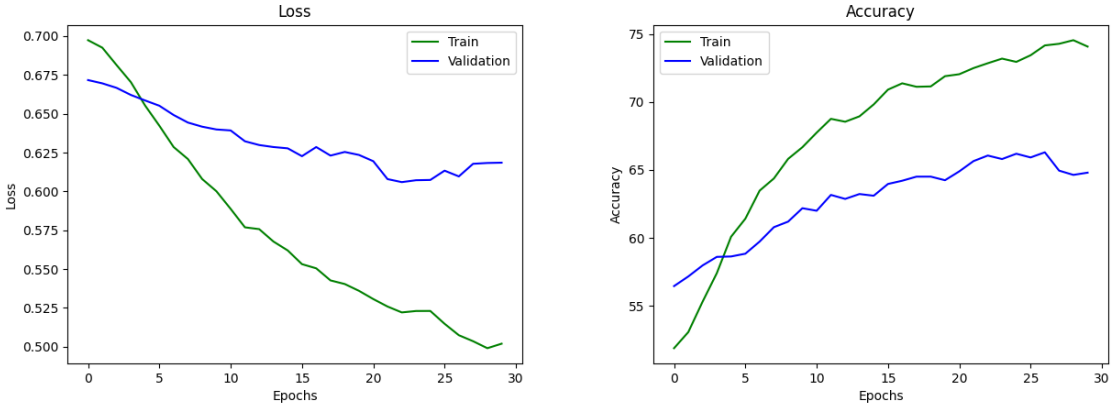

Figure 6.4: Loss and Accuracy graphics related with the train and validation process with the *SlowFast*, pre-trained with the *Kinetics 400*, and trained on the YNF dataset. These results were achieved through a process of 30 epochs, 0.0001 for learning rate, no data augmentation, with the optimizer SGD, batch size of 4, and frame rate of 2.

This section shows the results of the experiments made in the pre-trained models, intending to search for the optimal solution. Various experiments were made with different learning rates, sampling rates, and several epochs. The best results for the *SlowFast* model can be seen in Figure 6.4.

The left plot in Figure 6.4 shows the loss metric along the training and validation process with the *SlowFast*, pre-trained with the *Kinect 400*, and trained on the *YNF* dataset. The train and validation losses decrease along the process, achieving their lowest values in the 20th epoch. The training process was made for 50 epochs. After the 30th epoch, the validation loss raised, showing the beginning of overfitting.

The right plot in Figure 6.4 shows the accuracy metric for the training and validation process with the *SlowFast*. It can be observed that it achieved the highest validation accuracy value of approximately 65% in the 30th epoch.

B	0.87	0.13
NB	0.25	0.75
	B	**NB**

Figure 6.5: SlowFast Confusion Matrix

Figure 6.5 shows the percentage of true positive and false positive values achieved on the *YNF* dataset. It can be concluded that the model classifies the bullying actions with 87% of accuracy, and the non-bullying actions with 75% of accuracy. These values were achieved when used the threshold of 0.5 .

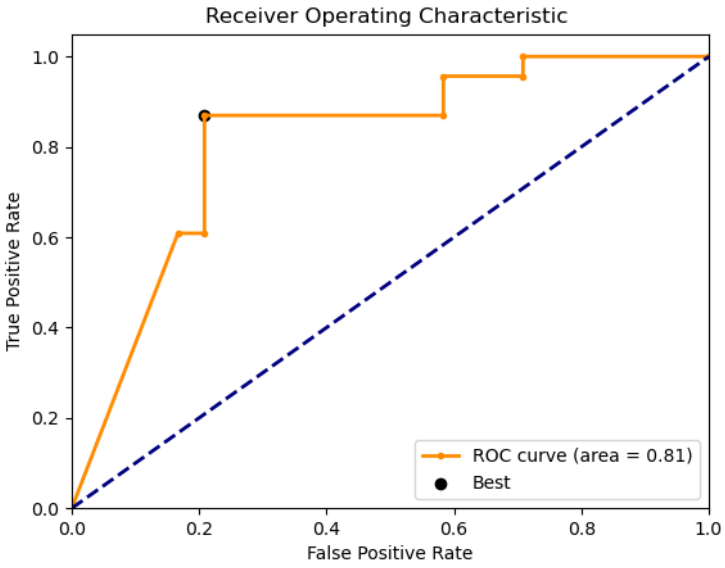

Figure 6.6: ROC Curve for the bullying class achieved with the *SlowFast* model. The best test threshold is annotated on the black dot.

Figure 6.6 shows the *Receiver Operating Characteristic* curve with a area value of 0.81. These metrics show the effectiveness of the model and it outstanding performance.
The next model fine-tuned with the *Kinetics 400* weights and trained with the *YNF* dataset was the ID3. The metrics along the training and validating process were collected to search for the optimal hyper-parameters. The left graphic on the figure 6.7 shows the decreasing loss achieved in training and validation processes with 50 epochs, 0.0001 for learning rate and frame rate of 6.

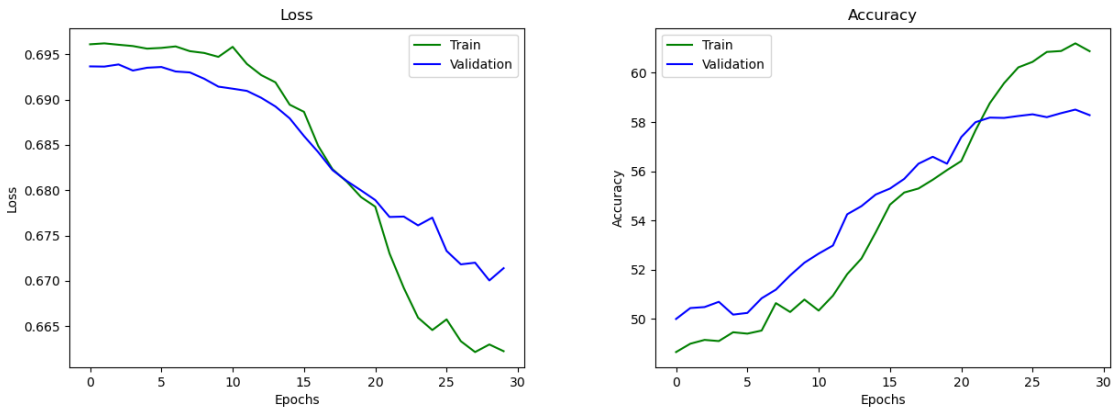

Figure 6.7: Loss and Accuracy graphics related with the train and validation process with the *I3D*, pre-trained with the *Kinetics 400* dataset weights and trained on the YNF dataset. These results were achieved through a process of 30 epochs, 0.0001 for learning rate, no data augmentation, SGD optimizer, batch size of 4, and frame rate of 6.

Analyzing Figure 6.7, the training and validating losses decreased along the process, achieving their lowest values in the 30th epoch. The training process was made for 50 epochs. After the 30th epoch, the validation loss raised, showing the beginning of overfitting.
The right graphic on the figure 6.7 shows the accuracy metric for the training and validat-

ing process with the *I3D* model. It can be observed that it achieved the highest validation accuracy value of approximately 59% in the 30th epoch.

B	0.83	0.17
NB	0.21	0.79
	B	NB

Figure 6.8: I3D Confusion Matrix

Figure 6.8 shows the percentage of the true positive and true negative values achieved on the *YNF* dataset. It can be concluded that the model classifies the bullying actions with 83% of accuracy, and the non-bullying actions with 79% of accuracy. These values were calculated considering the threshold of 0.5 .

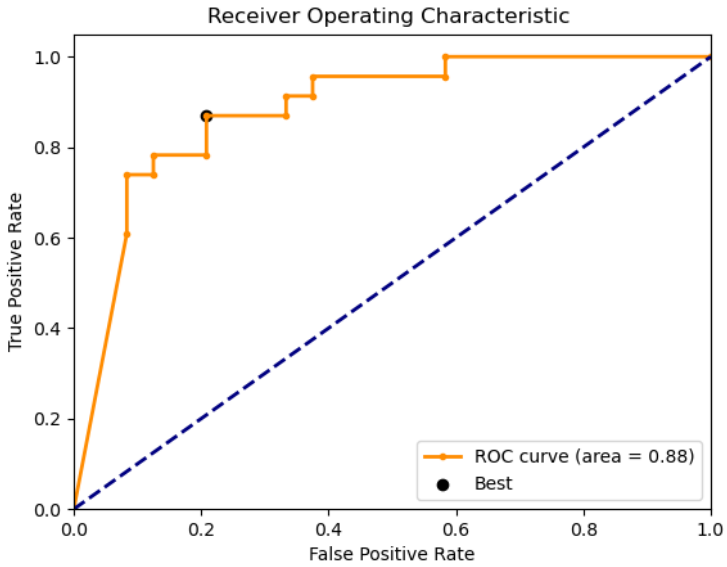

Figure 6.9: ROC Curve for the bullying class achieved with the *I3D* model. The black dot marked in the plot represent the best test threshold.

Figure 6.9 shows the *Receiver Operating Characteristic* curve with a area value of 0.88. These metrics show that this model achieved the best results so far and has a huge performance.

The last model trained with the *Kinetics 400* dataset weights through the fine-tuning process was the *C2D* architecture mentioned in Chapter 2. For this model, different experiments were made to search the hyper-parameters for the optimal solution. Figure 6.10 shows the training and validation process with the *C2D* trained on the *YNF* dataset, in 50 epochs, 0.0001 for learning rate, batch size of 4 and frame rate of 8.

The left plot in Figure 6.10 shows the loss metric along the training and validation process

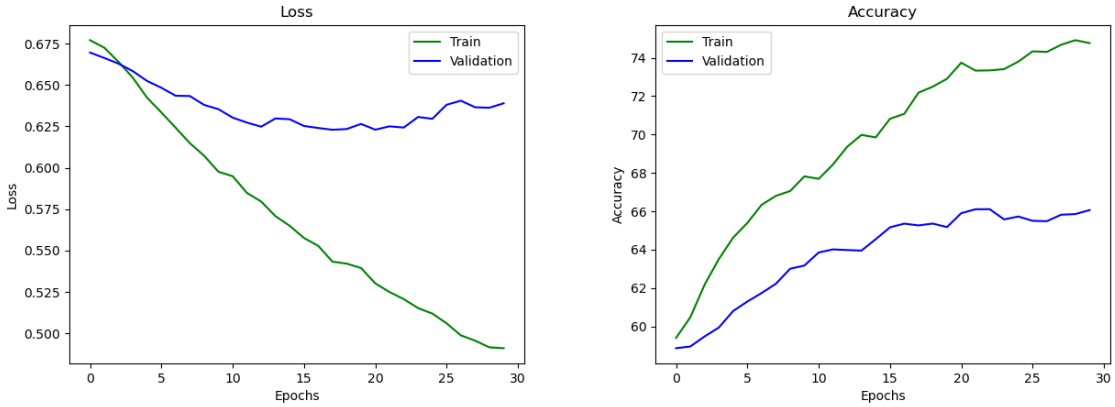

Figure 6.10: Loss and Accuracy graphics related with the train and validation process with the *C2D*, trained from scratch on the YNF dataset. These results were achieved through a process of 30 epochs, 0.0001 for learning rate, no data augmentation, SGD optimizer, batch size of 4, and frame rate of 8

with the *C2D*, pre-trained with the *Kinetics 400* dataset, and trained on the *YNF* dataset. Both the train and validation losses decrease along the process, achieving the lowest validation loss value in 20th epoch. The training process was made for 50 epochs, but after the 25th epoch, the validation loss raised, showing the beginning of overfitting.

The right plot in Figure 6.10 shows the accuracy metric for the training and validation process with the *C2D* model. It can be observed that it achieved the highest validation accuracy value of approximately 66% in the 25th epoch.

B	0.70	0.30
NB	0.17	0.83
	B	NB

Figure 6.11: C2D Confusion Matrix

Figure 6.11 shows the percentage of the true positive and false positives values achieved on the *YNF* dataset in test. It can be concluded that the model classifies the bullying actions with 70% of accuracy, and the non-bullying actions with 83% of accuracy. These metrics were calculated with a test threshold of 0.5 .

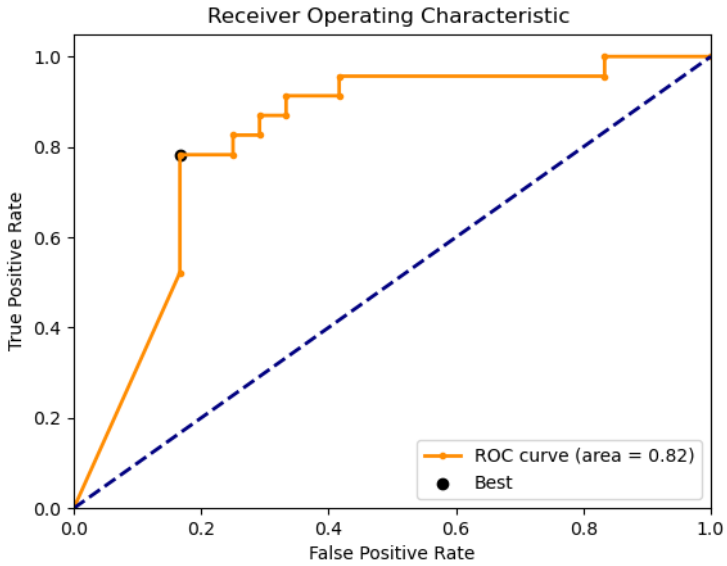

Figure 6.12: ROC Curve for the bullying class achieved with the *C2D* model. The black dot marked on the plot represent the best test threshold.

Figure 6.12 shows the *Receiver Operating Characteristic* curve area with value 0.82. These metrics show that the performance of the *C2D* model is similar of the *SlowFast* model, both with good outcomes.

Table 6.3 shows the final results achieved with all the models with different frame rates. Only the training and validation graphics related with the best sampling rate, for each model, were presented in this document.

YNF Dataset - Kinetics 400			
Sampling Rate	C2D (%)	I3D (%)	SlowFast (%)
2	72.34	72.34	82.98
4	70.21	78.72	80.85
6	72.34	80.85	76.60
8	76.60	70.21	76.60

Table 6.3: Top1-Accuracy on the test dataset achieved with the *SlowFast*, *I3D*, and *C2D* architectures trained on the *Kinetics 400* dataset and fine-tuned with the *YNF* dataset, with different sampling rates.

In analyze of Table 6.3, it can be concluded that so far, the best model for the task of classifying bullying on videos was the *SlowFast* architecture with an accuracy nearby 83 percent. These results were calculated using a threshold of 0.5. These results are not enough to conclude with determination which is the best model. Table 6.4 shows the last metrics calculated when the optimal threshold is calculated to reduce the number of false positives and raise the number of instances true positive.

YNF Dataset		
Model	Train Threshold	Accuracy
FGN	0.87	51.06%
SlowFast	0.11	82.98%
I3D	0.06	87.24%
C2D	0.52	76.60%

Table 6.4: Top1-Accuracy on test dataset, for the models presented in this document, with the optimal thresholds.

Table 6.4 is crucial to conclude with high certain the best model to perform the detecting bullying task. The train thresholds were calculated through the *ROC Curve* of the train dataset and further used on the calculation of the accuracy on the test dataset. This method give us a lot more knowledge about the best model to the task. The *FGN*, and *C2D* model accuracy were reduced when used the train threshold due to the low precision of the models. The other two models, *SlowFast* and *I3D* maintain the same accuracy because predicted the right class high confidence. This allows the conclusion that the *I3D* architecture is the best architecture for developing the application.

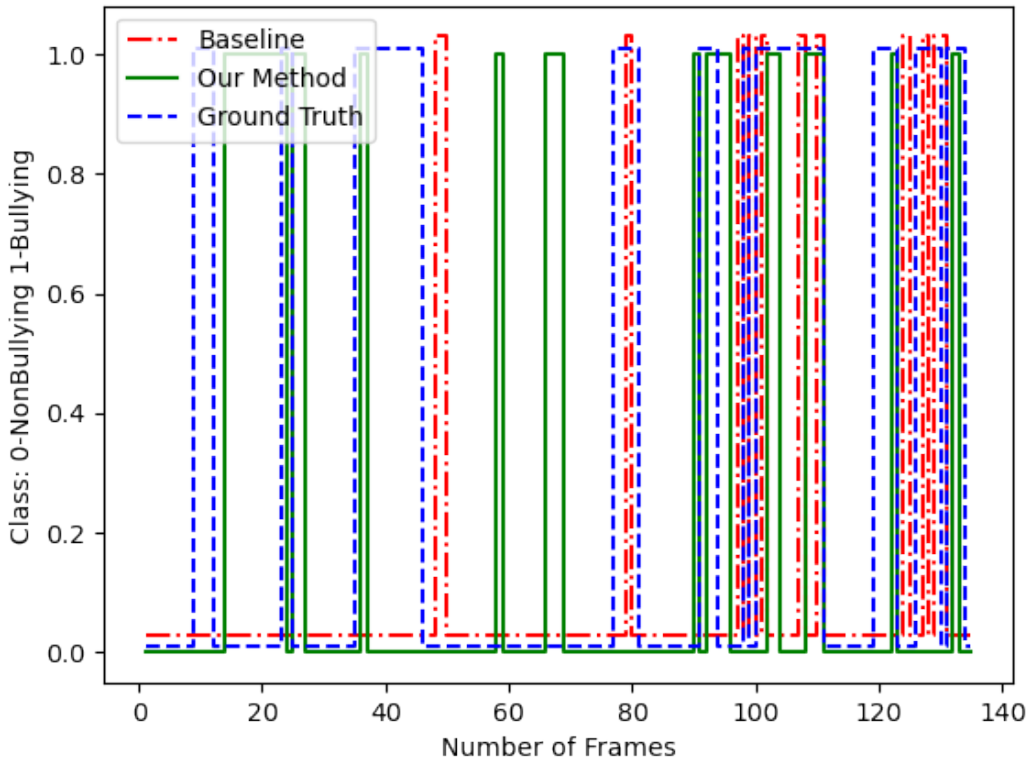

Figure 6.13: Inference plot of a demo bullying video that shows the ground truth in blue, the *I3D* model trained with the *YNF* dataset predictions in green, and the red predictions when used a model trained on the *RWF* dataset

.

Figure 6.13 shows the inference results achieved when using an untrimmed demo bullying video with the *I3D* model trained on the *YNF* dataset (green line), and a second model trained

on a fight dataset (red dash line). When both lines are compared, the ground truth (blue line), we conclude that the creation of an bullying dataset brought a huge improvement in the bullying detection task.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In conclusion, this work captured a novel bullying dataset, which classifies not only violent actions between students in school, but robbery, throwing of objects, and making fun of victims. Due to the difficult task of recording real bullying actions with young students, because of their inconstant behavior, the need for approval from the fathers, and the sensibility of the theme, the dataset is not rich in features and needs to be improved.
Besides that, the task of classifying actions on video was heavily studied; various models were used in the development phase with different methodologies. Only one model was trained with success from scratch due to the lower parameters number of the model. The other three models had to be trained with the *Kinetics 400* because no convergence was obtained when training them from scratch on the *YNF* dataset. With these three models, *C2D*, *I3D*, and *SlowFast*, it was possible to achieve the best result, 87.24 percent of accuracy on the *I3D* architecture considering the optimal threshold.
Due to the sensibility of the matter and the need of high guarantee of security for the schools and non-bullying organizations implement this solution in real world system, a study was made with the intention of reducing the number of resources needed to run this heavily applications without the need for sending data to big data centers, with possibility of leaking information along the way. For that purpose, a technique known as *Knowledge Distillation* was researched and deployed to improve the accuracy of a smaller model using knowledge acquired from a larger model. The tiny model can run on IoT and edge devices while maintaining the same accuracy as the teacher model. The ability to achieve good performance in models with minimal parameters enables their application in low-resource devices. Unfortunately, this technique could not produce results due to the necessity for massive disk memory to save the *Kinetics 400* dataset in numpy files with the optical flow and more than two weeks to create one experience to search for the optimal hyper-parameters.

7.2 Future Work

Recent research introduced a novel transformers model that produced the most outstanding results in various activities. In the future, one way to improve accuracy and efficiency could be to create a new transformer model without the need of huge computational resources.
Another improvement to this work is capturing more bullying videos with more students and other school environments to create a dataset rich in features and capable of detecting different actions. Acquiring actions from different countries is important because bullying actions change in other regions.

Some schools and non-bullying groups fear the implementation of these technologies due to the capturing and storing of videos obtained in private school properties containing the actions and faces of their kids. The use of IoT and edge devices is one of the methods that can provide another layer of security to the application since all of the computation required for application development is done without the use of the Internet or potentially vulnerable services. This ensures that data is stored in private school databases without needing extensive resources. To execute these deep learning models in such devices, network compression techniques such as pruning, quantization, and knowledge distillation are required to reduce model size while maintaining accuracy.

Bibliography

[1] R. Wang, "Incorporating frame image and frame sequence into ensemble learning networks to improve the accuracy of physical bullying-detecting model," in *IOP Conference Series: Materials Science and Engineering*, vol. 612, no. 5. IOP Publishing, 2019, p. 052047. xvii, xix, xx, xxiii, 15, 16, 17

[2] L. Ye, T. Liu, T. Han, H. Ferdinando, T. Seppänen, and E. Alasaarela, "Campus violence detection based on artificial intelligent interpretation of surveillance video sequences," *Remote Sensing*, vol. 13, p. 628, 2021. xvii, xx, 18, 19, 20

[3] Y. Xing, Y. Dai, K. Hirota, and Z. Jia, "A skeleton-based method for recognizing the campus violence," 2020. xvii, xx, xxiii, 20, 21, 22, 23

[4] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732. xix, 4

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, U. Eecs, A. Karpathy *et al.*, "C3d: Generic features for video analysis," in *2014 IEEE Conf. Comput. Vis. Pattern Recognit*, 2014, pp. 675–678. xix, 5

[6] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *Advances in neural information processing systems*, vol. 27, 2014. xix, 5, 6, 7

[7] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," *arXiv preprint arXiv:2106.11342*, 2021. xix, 6

[8] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634. xix, 7

[9] E. Voita, "NLP Course For You," Sep 2020. [Online]. Available: https://lena-voita.github.io/nlp_course.html xix, 8

[10] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803. xix, 9

[11] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308. xix, 9

[12] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6202–6211. xix, 10, 12

[13] M. Cheng, K. Cai, and M. Li, "Rwf-2000: an open large scale video database for violence detection," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 4183–4190. xix, 10, 11, 29

[14] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015. 12, 13

[15] C. Yuan and R. Pan, "Obtain dark knowledge via extended knowledge distillation," in *2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, 2019, pp. 502–508. 12

[16] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-second AAAI conference on artificial intelligence*, 2018. 21

[17] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 026–12 035. 21

[18] Fabian Caba. (2018) Activitynet. [Online]. Available: https://github.com/activitynet/ActivityNet/tree/master/Crawler/Kinetics 25

[19] (2021) Worldstarhiphop. [Online]. Available: https://worldstarhiphop.com/videos/ 26

[20] Cheng, Ming and Cai, Kunjing and Li, Ming. (2021) Rwf2000. [Online]. Available: https://github.com/mchengny/RWF2000-Video-Database-for-Violence-Detection 29

[21] H. Fan, Y. Li, B. Xiong, W.-Y. Lo, and C. Feichtenhofer, "Pyslowfast," https://github.com/facebookresearch/slowfast, 2020. 31, 32, 33

[22] Gao Peng . (2020) Cannot load c2d pre-trained model. [Online]. Available: https://github.com/facebookresearch/SlowFast/issues/163 32