UNIVERSITAT DE BARCELONA

Treball final de grau

GRAU DE INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Desenvolupament d'eines de IA pel Reconeixement de Menjar en Imatges

Autor: Junjie Li

Director: Dra. Petia Radeva i Dr. Marc Bolaños
Realitzat a: Departament de Matemàtiques i Informàtica.

Barcelona, 10 de Juny de 2024

Abstract

The rapid development of technology has significantly transformed various aspects of human life, including how we interact with food through digital platforms. This thesis focuses on two main objectives: developing an iOS Software Development Kit (SDK) for capturing data related to food for a model of food volume and quantity estimation, and enhancing an Image Transformer Network for recognition and classification of food.

In the first part of the thesis, we present the design and implementation of a powerful tool that allows users to capture data necessary for food volume and quantity estimation. Our method is implemented through a SDK collecting camera frame information, classifies, and processes these frames to gather accurate data about the food items. By analyzing the center of the scene and the angle formed between the first frame and the center, the necessary images are captured to support this estimation. We implemented various algorithms to dynamically detect, calculate, and update the scene center and to calculate the angles in real-time. These angles are categorized into different ranges to ensure the best data capture. After a detailed analysis of the system requirements, we conducted a comprehensive design. Key components of the SDK, including the acquisition and processing of camera frames, the interface design for sending data to the server, and the user interface design, are discussed. Through rigorous testing and optimization, we ensured the performance and usability of the SDK.

The second part of the thesis builds on previous research to optimize the accurate recognition and classification of food using an Image Transformer Network. A deep learning-based visual recognition model was trained on a diverse and extensive food image dataset. We explored various model architectures and optimization strategies to improve the accuracy and efficiency of the recognition system. This process included data collection and preprocessing, model selection, training methods, and hyperparameter tuning. The trained models were then evaluated against the food classification dataset owned by LogMeal. Finally, we discussed the deployment and practical application of the models in real-world scenarios.

By combining these two innovative approaches, this thesis provides a comprehensive solution for digital food data capture and recognition. It makes contributions to mobile application development and Image Transformer Network-based image recognition. The integration of the SDK with the visual recognition model opens up new avenues for applications in food identification, health, dining experiences, and diet management.

Resume

El rápido desarrollo de la tecnología ha transformado significativamente varios aspectos de la vida humana, incluida nuestra interacción con los alimentos a través de plataformas digitales. Esta tesis se centra en dos objetivos principales: desarrollar un Software Development Kit (SDK) para iOS para capturar datos relacionados con los alimentos para un modelo de estimación de volumen y cantidad de alimentos, y mejorar una Image Transformer Network para el reconocimiento y clasificación de alimentos.

En la primera parte de la tesis, presentamos el diseño y la implementación de una herramienta poderosa que permite a los usuarios capturar datos necesarios para la estimación del volumen y la cantidad de alimentos. Nuestro método se implementa a través de un SDK que recopila información de los fotogramas de la cámara, clasifica y procesa estos fotogramas para recopilar datos precisos sobre los alimentos. Al analizar el centro de la escena y el ángulo formado entre el primer fotograma y el centro, se capturan las imágenes necesarias para apoyar esta estimación. Implementamos varios algoritmos para detectar, calcular y actualizar dinámicamente el centro de la escena y para calcular los ángulos en tiempo real. Estos ángulos se categorizan en diferentes rangos para asegurar la mejor captura de datos. Después de un análisis detallado de los requisitos del sistema, realizamos un diseño integral. Se discuten los componentes clave del SDK, incluida la adquisición y el procesamiento de fotogramas de la cámara, el diseño de la interfaz para enviar datos al servidor y el diseño de la interfaz de usuario. A través de pruebas rigurosas y optimización, aseguramos el rendimiento y la usabilidad del SDK.

La segunda parte de la tesis se basa en investigaciones previas para optimizar el reconocimiento y la clasificación precisa de los alimentos utilizando una Image Transformer Network. Se entrenó un modelo de reconocimiento visual basado en aprendizaje profundo en un conjunto de datos de imágenes de alimentos diverso y extenso. Exploramos varias arquitecturas de modelos y estrategias de optimización para mejorar la precisión y la eficiencia del sistema de reconocimiento. Este proceso incluyó la recopilación y preprocesamiento de datos, la selección de modelos, los métodos de entrenamiento y el ajuste de hiperparámetros. Los modelos entrenados se evaluaron frente al conjunto de datos de clasificación de alimentos propiedad de LogMeal. Finalmente, discutimos el despliegue y la aplicación práctica de los modelos en escenarios del mundo real.

Combinando estos dos enfoques innovadores, esta tesis proporciona una solución integral para la captura y el reconocimiento digital de datos alimentarios. Contribuye al desarrollo de aplicaciones móviles y al reconocimiento de imágenes basado en redes de transformadores de imágenes. La integración del SDK con el modelo de reconocimiento visual abre nuevas vías para aplicaciones en identificación de alimentos, salud, experiencias gastronómicas y gestión de dietas.

Resum

El ràpid desenvolupament de la tecnologia ha transformat significativament diversos aspectes de la vida humana, incloent-hi la nostra interacció amb els aliments a través de plataformes digitals. Aquesta tesi se centra en dos objectius principals: desenvolupar un Software Development Kit (SDK) per a iOS per capturar dades relacionades amb els aliments per a un model d'estimació de volum i quantitat d'aliments, i millorar una Image Transformer Network per al reconeixement i classificació d'aliments.

En la primera part de la tesi, presentem el disseny i la implementació d'una eina poderosa que permet als usuaris capturar dades necessàries per a l'estimació del volum i la quantitat d'aliments. El nostre mètode s'implementa mitjançant un SDK que recull informació dels fotogrames de la càmera, classifica i processa aquests fotogrames per recopilar dades precisos sobre els aliments. En analitzar el centre de l'escena i l'angle format entre el primer fotograma i el centre, es capturen les imatges necessàries per donar suport a aquest estimació. Vam implementar diversos algoritmes per detectar, calcular i actualitzar dinàmicament el centre de l'escena i per calcular els angles en temps real. Aquests angles es categoritzen en diferents rangs per assegurar la millor captura de dades. Després d'una anàlisi detallada dels requisits del sistema, vam dur a terme un disseny integral. Es discuteixen els components clau del SDK, inclosa l'adquisició i el processament de fotogrames de la càmera, el disseny de la interfície per enviar dades al servidor i el disseny de la interfície d'usuari. A través de proves rigoroses i optimització, vam assegurar el rendiment i la usabilitat del SDK.

La segona part de la tesi es basa en investigacions prèvies per optimitzar el reconeixement i la diferenciació precisa dels aliments utilitzant una Image Transformer Network. Es va entrenar un model de reconeixement visual basat en aprenentatge profund en un conjunt de dades d'imatges d'aliments divers i extens. Vam explorar diverses arquitectures de models i estratègies d'optimització per millorar la precisió i l'eficiència del sistema de reconeixement. Aquest procés va incloure la recopilació i preprocessament de dades, la selecció de models, els mètodes d'entrenament i l'ajust d'hiperparàmetres. Els models entrenats es van avaluar enfront del conjunt de dades de classificació d'aliments propietat de LogMeal. Finalment, vam discutir el desplegament i l'aplicació pràctica dels models en escenaris del món real.

Combinant aquests dos enfocaments innovadors, aquesta tesi proporciona una solució integral per a la captura i el reconeixement digital de dades alimentàries. Contribueix al desenvolupament d'aplicacions mòbils i al reconeixement d'imatges basat en xarxes de transformadors d'imatges. La integració del SDK amb el model de reconeixement visual obre noves vies per a aplicacions en identificació d'aliments, salut, experiències gastronòmiques i gestió de dietes.

Agraïments

Vull expressar el meu sincer agraïment als meus tutors de treball de fi de grau, la Dra. Petia Radeva i el Dr. Marc Bolaños, per la seva orientació i suport constant al llarg de tot aquest projecte. Les seves idees i consells han estat fonamentals per a aquest treball.

També vull agrair a Leonel Viera per la seva excel·lent guia i ajuda en el desenvolupament del SDK. La seva experiència i coneixement han estat inestimables per superar molts reptes tècnics.

A més, estic profundament agraït a Maria Fernanda Herrera per la seva ajuda en la resolució de qualsevol problema que sorgís, així com pel seu suport en les proves i els algorismes del SDK. La seva col·laboració ha estat clau d'aquest projecte.

També vull reconèixer l'ajuda d'Andrey Nunez en el desenvolupament dels models de reconeixement visual. La seva experiència i treball en aquest àmbit ha estat vital per millorar la precisió i l'eficàcia dels nostres resultats.

Finalment, vull expressar el meu agraïment a tot l'equip de LogMeal pel seu suport continuat i per l'experiència inestimable que he adquirit treballant amb ells.

Índex

Capítol 1

Introducció

Aquest treball de fi de grau es desenvolupa en col·laboració amb LogMeal Food AI[1], una empresa en tecnologies d'Intel·ligència Artificial aplicades a la indústria alimentària. El treball ha estat supervisat pels meus tutors de la Universitat de Barcelona, la Dra. Petia Radeva i el Dr. Marc Bolaños, que també són cofundadors de LogMeal. Les seves experteses i visions han estat fonamentals per guiar aquest treball cap a solucions innovadores i pràctiques.

El focus del treball es divideix en dues parts principals, ambdues estretament lligades a la col·laboració amb LogMeal:

1. **Desenvolupament d'un SDK per a iOS per a la captura de dades de menjar per a models d'estimació de volum i quantitat de menjar**: En aquesta primera part, es presenta el disseny i implementació d'un Software Development Kit[2] (SDK) que permet als usuaris capturar dades relacionades amb el menjar per a models d'estimació del volum i la quantitat de menjar. Aquest SDK utilitza algoritmes avançats de processament d'imatges per analitzar els fotogrames capturats per la càmera del dispositiu, recollint dades precises sobre els aliments.
2. **Millora d'una xarxa neuronal Image Transformer Network[3] per al reconeixement visual i classificació del menjar**: La segona part del treball se centra en l'optimització d'una xarxa neuronal *Image Transformer Network* dedicada al reconeixement i classificació del menjar. S'han explorat diverses arquitectures de models i estratègies d'optimització, utilitzant un conjunt de dades d'imatges de menjar per entrenar el model.

Aquest treball posa en relleu la sinergia entre la recerca acadèmica i la innovació empresarial, demostrant com la col·laboració amb LogMeal ha estat clau per al desenvolupament de tecnologies avançades en la digitalització de la gastronomia.

[1] `https://logmeal.com/en/`

[2] SDK és un conjunt d'eines que facilita als desenvolupadors la creació d'aplicacions per a una plataforma específica.

[3] Image Transformer Network és un tipus de xarxa neuronal que utilitza l'arquitectura Transformer, originàriament desenvolupada per al processament de seqüències de text, per a tasques de visió per computador, com el reconeixement d'imatges.

1.1 Context

LogMeal és una aplicació d'intel·ligència artificial dissenyada per reconèixer aliments i proporcionar informació nutricional a partir d'una foto presa per l'usuari. L'objectiu de LogMeal és simplificar i automatitzar el procés de càlcul precís de la quantitat d'aliments, oferint una informació nutricional detallada. Aquesta eina utilitza tecnologies com *Deep Learning*[1] i algoritmes de visió per computador en l'àmbit alimentari.

LogMeal va néixer l'any 2012 com a spin-off[4] de la Universitat de Barcelona (UB)[5]. L'aplicació permet analitzar en temps real els àpats amb una simple fotografia. Detecta cada aliment al plat o safata, avalua la quantitat servida i estima tots els valors nutricionals clau, tant micro com macro. LogMeal ofereix l'API més avançada en intel·ligència artificial alimentària, seguiment d'aliments i generació de dietaris. És la solució ideal per a empreses que necessiten informació detallada d'imatges d'aliments, incloent-hi tipus d'aliment, grups d'aliments, plats, ingredients o receptes, així com informació nutricional (32 nutrients, micro i macro). Aquesta API està disponible en 35 llenguatges de programació, proporcionant una solució completa per al seguiment exhaustiu de clients, esportistes o pacients.

Els serveis que ofereix LogMeal inclouen LogMeal API, LogMeal APP, LogMeal PLATFORM i LogMeal KIOSK, ho explicarem a continuació:

- **LogMeal API**: Aquesta API està dissenyada per al reconeixement d'aliments i la detecció d'aliments basada en imatges. Ofereix etiquetatge semàntic, identificant grups d'aliments, plats i ingredients, a més d'analitzar la informació nutricional. L'API permet accedir a les funcionalitats i algoritmes d'intel·ligència artificial, visió per computador i deep learning desenvolupats per LogMeal.

Figure 1.1: Logmeal API. Font: LogMeal, 2017, LogMeal Food AI[6].

- **LogMeal APP**: L'aplicació de LogMeal per a dispositius Android i iOS permet als usuaris ser més conscients de les seves ingestes, ajudant-los a mantenir una dieta saludable i a millorar la seva qualitat de vida. Utilitza LogMeal API; funciona d'aquesta manera: amb una simple foto del teu àpat, LogMeal el reconeixerà en temps real i proporcionarà:
  - Llista d'ingredients: Presenta els ingredients del plat amb quantitats estàndard, que es poden ajustar, recalculant automàticament tots els valors.
  - Informació nutricional detallada: Proporciona 32 indicadors nutricionals com energia, macronutrients, fibra, colesterol, vitamines i minerals.

[4] spin-off és una nova empresa o projecte que es desenvolupa a partir d'una empresa o projecte existent(UB).

[5] `http://www.fbg.ub.edu/es/actualidad/nace-aigecko-technologies-inteligencia-artificial-servicio-reconocimiento-imagenes/`

[6] `https://api.logmeal.com/docs/`

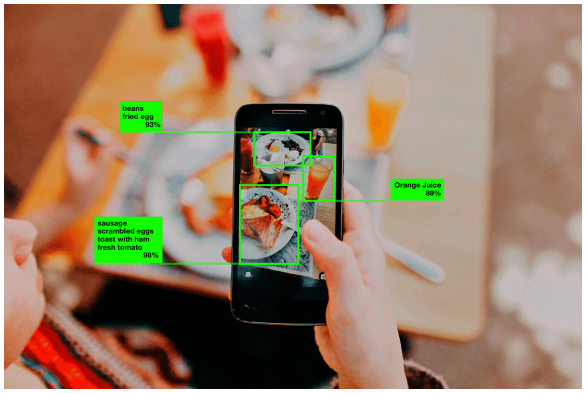

Figure 1.2: LogMea APP. Font: LogMeal, 2017, LogMeal Food AI[7].

- **LogMeal PLATFORM**: Aquesta eina web està dissenyada per a experts com nutricionistes i sanitaris. Permet establir objectius nutricionals, gestionar, seguir i monitoritzar dades, i visualitzar alertes i indicadors. Ofereix gràfics detallats d'indicadors de nutrients, visualització de rangs de dades, i un resum de la ingesta d'un usuari.

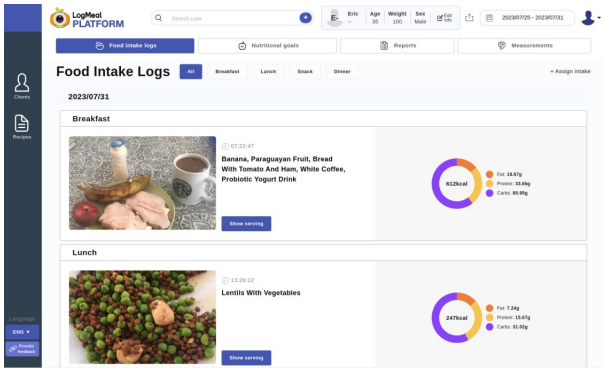

Figure 1.3: LogMeal PLATFORM. Font: Font: LogMeal, 2017, LogMeal Food AI[8].

- **LogMeal KIOSK** Fig. 1.4: Integra el programari de LogMeal en quioscs de socis fabricants, facilitant la seva implementació en qualsevol restaurant o establiment. El funcionament és senzill: el client col·loca la seva safata amb els aliments al quiosc, que pren una foto i utilitza algoritmes al núvol per reconèixer cada aliment en temps real. El sistema és adaptable a qualsevol restaurant, aprenent els aliments servits en temps real, detectant aliments, preus, combinacions de menús i oferint solucions de pagament per al client.

En aquest context, LogMeal aspira a ser un pioner en el camp de la identificació d'aliments i l'anàlisi nutricional a nivell d'aplicació. Aquest treball es centra principalment en el desenvolupament d'un SDK per a LogMeal APP compatible amb iOS, i es preveu la seva integració en un llenguatge de programació de front-end mòbil, React-native[10]. Amb aquest SDK acabat s'integrarà a l'API de LogMeal. I l'API de LogMeal també forma part del nostre projecte, opti-

[7] `https://apps.apple.com/es/app/logmeal/id1578738525`
[8] `https://logmeal.com/en/nutra-expert`
[9] `https://logmeal.com/kiosk`
[10] `https://reactnative.dev/`

Figure 1.4: LogMeal KIOSK. Font: LogMeal, 2017, LogMeal Food AI[9].

mitzarem els models d'intel·ligència artificial basats en Image Transformer Network per a una millor integració amb l'SDK i obtenir els resultats més òptims.

Aquest projecte té una importància vital no només per a LogMeal en si mateix, sinó també per a qualsevol persona o empresa interessada en utilitzar aquest SDK i API. La planificació d'aquest treball és ambiciosa i té un gran impacte en LogMeal.

Figure 1.5: Logotip de LogMea. Font: LogMeal, 2017, LogMeal Food AI.

1.2 Motivació

Aquest projecte té com a motivació l'ús de plataformes digitals i tecnologies d'intel·ligència artificial per proporcionar a l'usuari eines clau per estimar amb precisió la ingesta d'aliments, millorant significativament la qualitat de vida. Segons l'Organització Mundial de la Salut (OMS), una de cada sis malalties està directament relacionada amb la dieta, destacant així la urgència de solucionar els problemes de salut[2].

Els mètodes tradicionals d'avaluació nutricional sovint es veuen limitats per la intervenció humana, podent resultar en dades no precises o obsoletes. Amb la innovació de les plataformes digitals i la intel·ligència artificial, podem superar aquests reptes i proporcionar als usuaris anàlisis i consells nutricionals personalitzats i en temps real.

Establir hàbits alimentaris flexibles és clau per reduir el risc de problemes de salut. Les nostres eines no només ajuden els usuaris a estimar amb precisió la ingesta d'aliments, sinó que també ofereixen plans de dieta personalitzats basats en les seves preferències i necessitats nutricionals

individuals. Gràcies a l'aprenentatge i a l'optimització continuada dels algoritmes intel·ligents, podem proporcionar consells nutricionals cada cop més precisos i efectius.

Els mals hàbits alimentaris poden conduir a diverses malalties, com ara diabetis, osteoporosi, malalties cardiovasculars, entre altres. Mitjançant l'ús d'eines de valoració nutricional i estimació de la ingesta, podem ajudar els usuaris a prevenir aquestes malalties. A més, oferim consells de salut personalitzats, com ara controlar la ingesta calòrica per cada àpat o augmentar la ingesta de fibra i vitamines, per ajudar els usuaris a mantenir una bona salut i reduir el risc de malalties[3].

A més, la nostra plataforma proporciona avaluacions de riscos de salut i mesures preventives. Mitjançant l'anàlisi de dades alimentàries dels usuaris, podem identificar riscos potencials per a la salut i proporcionar consells de gestió de la salut personalitzats. Amb intervencions i orientacions oportunes, ajudem els usuaris a millorar el seu estil de vida, reduint així el risc de malalties i millorant la seva qualitat de vida.

A més dels beneficis individuals, l'ús de plataformes digitals i tecnologies d'intel·ligència artificial en la gestió nutricional també pot tenir un impacte positiu en la salut pública. L'anàlisi de dades agregades i anonimitzades dels usuaris pot oferir informació valuosa per a investigacions epidemiològiques, permetent identificar tendències alimentàries, factors de risc poblacionals i àrees amb necessitats d'intervenció nutricional específica. Aquesta informació pot ajudar als responsables de la salut pública a desenvolupar polítiques i programes més eficaços per a la promoció de dietes saludables a gran escala. A més, la col·laboració amb professionals de la salut i institucions educatives pot ampliar l'abast i l'eficàcia de les nostres eines, contribuint a la creació d'una comunitat més conscient i saludable.

1.3 Objectius

Aquest treball té com a objectiu principal dues parts:

Primera part: Desenvolupament i validació d'un model de l'estimació de la quantitat de menjar (food volume i food quantity estimation) implementant el model en un SDK per a iOS.

En aquesta primera part, es presenta el disseny i implementació d'un Software Development Kit (SDK) que permet als usuaris capturar dades relacionades amb el menjar per a models d'estimació de volum i quantitat de menjar. Aquest SDK utilitza algoritmes avançats de processament d'imatges per analitzar els fotogrames capturats per la càmera del dispositiu, recollint dades precises sobre els aliments. Els nostres objectius inclouen:

1. Implementar diversos algorismes per a la detecció dinàmica, el càlcul en temps real i l'actualització del centre de l'escena.
2. Calcular l'angle entre el primer fotograma i el centre per capturar les imatges necessàries per a la representació del volum i la quantitat d'aliments, classificant-les en diferents rangs.

3. Dissenyar els components clau del SDK, incloent la captura i el processament dels fotogrames de la càmera, el disseny de la interfície d'usuari, i el disseny de l'API per a l'enviament de dades al servidor, amb l'objectiu de facilitar la integració ràpida i l'ús pels desenvolupadors.

4. Assegurar el rendiment i la disponibilitat del SDK mitjançant proves rigoroses i optimitzacions.

Segona part: Millora d'una xarxa neuronal *Image Transformer Network* per al reconeixement i classificació del menjar

Aquesta part es centra en optimitzar una xarxa neuronal Image Transformer Network per al reconeixement i classificació d'aliments. Basant-nos en investigacions anteriors, utilitzarem mètodes com la recopilació i el pre-processament de dades, la selecció de models, tècniques d'entrenament i ajust de paràmetres, i explorarem diverses arquitectures i estratègies d'optimització per millorar la precisió i l'eficiència del sistema de reconeixement. Finalment, avaluarem el model entrenat segons el conjunt de dades de classificació d'aliments de LogMeal.

A més, aquest treball destaca la col·laboració entre la investigació acadèmica i la innovació comercial, que demostra una solució integral per a la visualització i identificació digital d'aliments utilitzant aplicacions mòbils i intel·ligència artificial.

1.4 Organització

En les seccions anteriors, hem donat una breu introducció al context del projecte i hem detallat els objectius de les dues parts principals. Ara, explicarem detalladament com hem aconseguit durant l'últim any aquests objectius i plans.

Al capítol dos, ens centrarem en la primera part del projecte, és a dir, l'anàlisi, el disseny i la implementació del mètode d'estimació de la qüamtita del menjar a partir d'imatges implementant-lo con un SDK. Explicarem en detall els processos d'anàlisi, disseny i implementació, i conclourem amb els resultats finals després de les proves. Aquesta secció proporcionarà una comprensió completa del desenvolupament del SDK.

El capítol tres se centrarà en la segona part del projecte. Com aquesta part és relativament independent de la primera, en primer lloc repassarem les investigacions anteriors i en farem una anàlisi. A continuació, reproduirem aquestes investigacions i explorarem com optimitzar i entrenar una Image Transformer Network. Finalment, analitzarem els resultats obtinguts i avaluarem si hem assolit els objectius esperats.

Al final del projecte, a més de resumir els resultats obtinguts, farem recomanacions per a futurs treballs, amb l'objectiu que pugui tenir una comprensió clara de la progressió global del projecte. També destaquem la importància del projecte i les seves possibles aplicacions perquè entengui millor d'aquest treball.

1.5 Planificació

Per a poder avançar eficaçment i mostrar els resultats del nostre projecte de la millor manera possible, hem dut a terme dues parts principals durant l'últim any. A continuació, us presentem una planificació detallada:

Part 1: Desenvolupament del SDK d'iOS. *Des de setembre del 2023 fins a març del 2024*

Orientació i formació inicial: Al començar el projecte, ens vam posar en contacte amb *Leonel Viera*, director de Panalsoft[11]. Leo ens va ajudar a familiaritzar-nos amb el projecte i a completar les tasques inicials.

Mètode de treball i col·laboració en equip: El nostre mètode de treball consisteix a descompondre les grans tasques del SDK en mòduls més petits, analitzar-los, provar-los i, finalment, integrar-los pas a pas. Això ens permet abordar els problemes amb més facilitat i també permet als membres de l'equip completar les seves tasques de manera independent.

Reunions setmanals: Des que vam iniciar el projecte a finals de setembre del 2023 fins a finals de febrer del 2024, hem participat en reunió setmanal d'aproximadament una hora els dimarts. Els participants a la reunió inclouen:

- Marc Bolaños (CTO i cofundador).
- Eric Verdaguer (CEO i cofundador).
- Leonel Viera (Director del Panalsoft).
- Un company en el desenvolupament del SDK d'Android.

Durant les reunions, discutim les tasques completades, compartim els problemes i busquem solucions conjuntes, i programem les tasques per al futur. A més, en el cas de problemes tècnics i proves, contactem amb la *Maria Fernanda*[12], qui ens ha donat un suport important en la implementació dels algoritmes.

Part 2: Optimització de Image Transformer Network. *Des març del 2024 fins a maig del 2024*

Descripció de les tasques: La segona part del projecte consisteix a optimitzar una xarxa neuronal de Image Transformer Network per al reconeixement i classificació d'aliments, basant-nos en la recerca anterior. En les fases inicials de la tasca, hem dedicat molt temps a analitzar l'article d'*Andrey Nunez* i del *CSWin Transformer*[13]. Després, hem començat a explorar diverses arquitectures de xarxes i estratègies d'optimització per millorar el resultat de la xarxa de reconeixement.

Mètode de treball: Les tasques de la segona part tendeixen a ser més independents. El nostre mètode de treball inclou:

[11] Una empresa col·laboradora de LogMeal en el desenvolupament de APP. `https://panalsoft.com/`

[12] Experta en processament d'imatges i visió computador

[13] CSWin Transformer: presenta una nova arquitectura de Transformer per a l'extracció i el processament de característiques en tasques visuals.

- **Investiga articles**: Investigar i analitzar detalladament treball d'Andrey, comprendre els articles, reproduir els resultats i buscar punts d'optimització potencials.
- **Comunicació tècnica**: Tot i que les tasques són més independents, però mantenim comunicació regular amb els membres de l'equip i els experts, compartint els avenços de la investigació i els problemes trobats, i rebre consells i retroalimentació.
- **Registre documental**: Registrar detalladament el procés d'investigar, els resultats dels experiments i les mesures d'optimització a cada etapa, per a futures investigacions i projectes.

Reunions setmanals: Com vaig comentar dalt, des de principis de març del 2024 fins a finals de maig del 2024, fem reunió de seguiment setmanal cada dilluns amb:

- Dr. Marc Bolaños (CTO, cofundador i Expert de Visió Computador)
- Maria Fernanda Herrera (Experta en processament d'imatges i Visió Computador)

A més, també ens posem en contacte amb *Andrey Nunez*, qui ens ajuda com a expert de l'indústria.

Seguiment detallat del treball

Per fer un seguiment detallat de les tasques i del progrés del treball de fi de grau, realitzem múltiples reunions amb Petia Radeva i Marc Bolaños, on discutim les tasques completades i els pendents. En aquestes reunions, Petia Radeva aporta opinions i metodologies per ajudar-nos a assolir millor els objectius.

Gràfic de progrés de tasques

Finalment, hem proporcionat un diagrama de Gantt Fig. 1.6 que mostra clarament la durada de totes les tasques i les seves dependències entre elles.

A través d'aquest diagrama, presentem de manera clara i estructurada les diferents parts del projecte i els progressos de cada setmana al llarg de tot l'any. Hem dividit l'any en setmanes i hem subdividit el projecte en dues parts, cadascuna amb les seves pròpies tasques i els objectius assolits en cada fase. Aquesta representació visual exhaustiva ens permet planificar i gestionar millor el progrés del projecte, assegurant que les tasques de cada fase es compleixin puntualment i garantint el desenvolupament del projecte i la seva entrega.

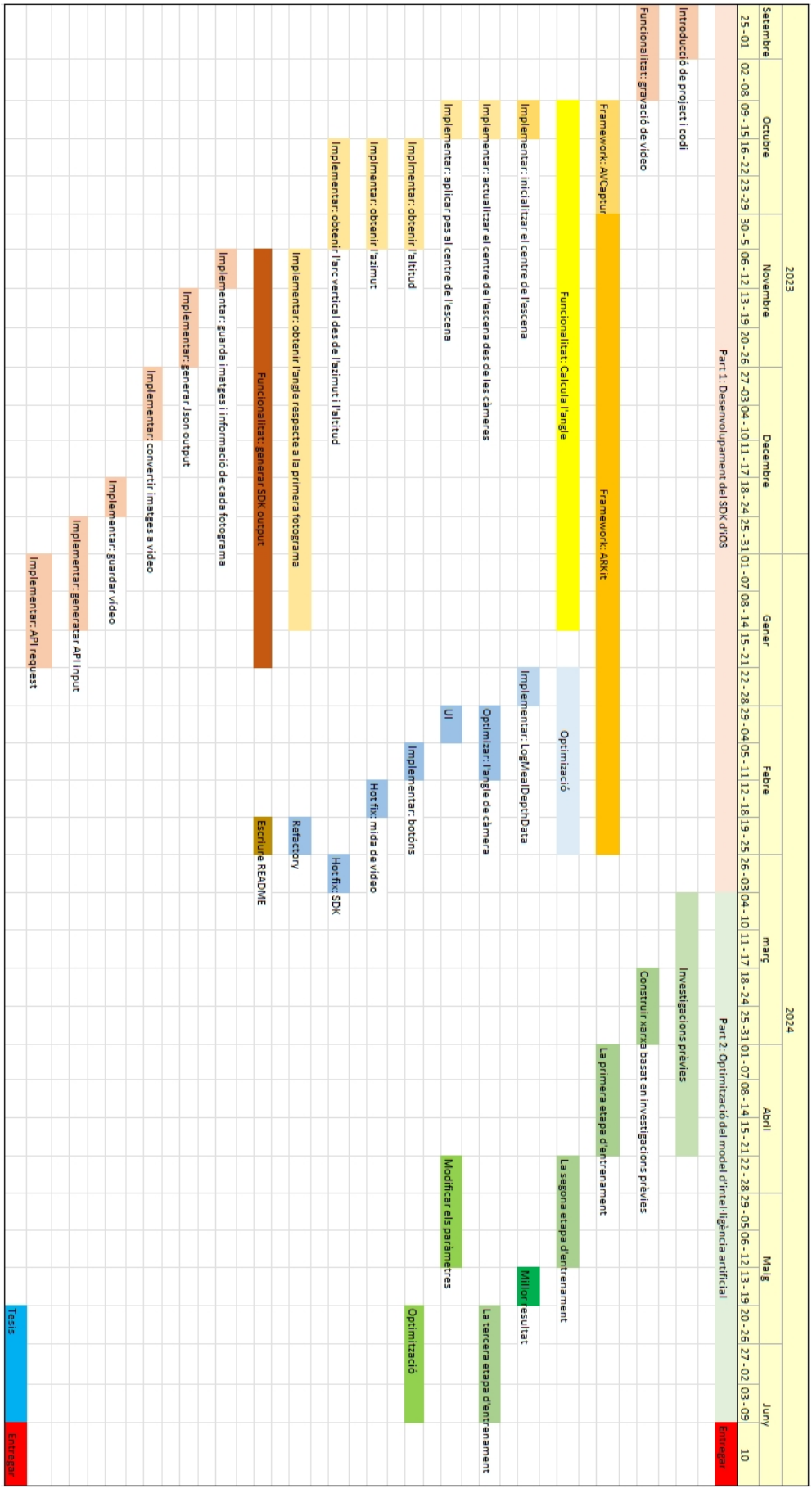

Figure 1.6: Diagrama de Gantt

Capítol 2

Part1: Disseny i implementació del mètode d'estmació del volum del menjar

En aquest capítol, començarem la primera part del nostre projecte, que consisteix en el disseny i la implementació del mètode d'estimació de la quantitat del menjar imaplementant-lo via SDK per a iOS. Explicarem detalladament el procés d'anàlisi, disseny i implementació del programa.

Primer, hem d'entendre els requisits importants del projecte i analitzar-los des de tres perspectives: requisits funcionals, requisits no funcionals i requisits dels usuaris. Els requisits funcionals especifiquen els serveis que l'SDK ha de proporcionar i els seus objectius; els requisits no funcionals analitzen l'equilibri entre els requisits funcionals i la usabilitat per assegurar-se que els usuaris puguin executar les operacions de captura sense problemes; i els requisits dels usuaris se centren a oferir una interfície intuïtiva i fàcil d'usar.

A continuació, per comprendre completament el funcionament de l'aplicació, discutirem un diagrama de flux, que és una representació gràfica d'un algorisme o procés. En el diagrama de flux, explicarem detalladament cada pas, des del moment en què l'usuari decideix començar a capturar imatges fins que les imatges s'envien a l'API.

Després, mostrarem el diagrama de l'estructura del codi del projecte i explicarem breument per què s'ha de construir d'aquesta manera. Aquesta estructura ajuda a millorar la mantenibilitat i extensibilitat del projecte.

Pel que fa al disseny de la interfície d'usuari, el nostre objectiu és assegurar-nos que els usuaris sempre coneguin l'estat real de la captura. Per això, hem realitzat una investigació per optimitzar la intuïció i la facilitat d'ús de la interfície.

Seguidament, discutirem detalladament els aspectes tècnics de la implementació, incloent com obtenir les dades dels fotogrames, calcular la posició de la càmera i la posició central de l'escena, calcular l'angle de la càmera i com generar l'arxiu de resultats per enviar-lo al servidor de l'API LogMeal per als models de volum de menjar i estimació de quantitat de menjar.

Finalment, analitzarem els resultats generats per l'SDK i els sotmetrem a proves exhaustives per assegurar-ne l'exactitud i la fiabilitat.

A través d'aquests passos, entendrem i implementarem completament el disseny i la implementació de l'SDK per a iOS, oferint als usuaris una solució eficient i fàcil d'usar.

2.1 Anàlisi de requisits

En el desenvolupament de l'SDK d'iOS, els dispositius de prova han de tenir característiques específiques, ja que no tots els dispositius iOS són adequats. L'SDK utilitza una plataforma anomenada ARKit. iOS ARKit[4] és un framework de realitat augmentada (AR) desenvolupat per Apple, utilitzat per crear experiències AR immersives en dispositius iOS. Per utilitzar ARKit, els dispositius han de complir els següents requisits:

Requisits del sistema

1. Sistema operatiu: el dispositiu ha de funcionar amb iOS 11 o una versió posterior.
2. Processador: el dispositiu ha d'estar equipat amb un processador A9 o una versió posterior.

En concret, els següents dispositius són compatibles amb ARKit:

Dispositius compatibles amb ARKit:

- iPhone
  - iPhone 6s i 6s Plus (i tots els models posteriors)
- iPad
  - iPad (5a generació i tots els models posteriors)
  - iPad mini (5a generació i tots els models posteriors)
  - iPad Air (3a generació i tots els models posteriors)
  - iPad Pro (tots els models, els models equipats amb un escàner LiDAR[1] i posteriors)

Dispositius recomanats

Per obtenir la millor experiència amb ARKit, es recomana utilitzar dispositius equipats amb el xip A12 Bionic o una versió posterior. Aquests dispositius tenen una capacitat de processament més potent i sensors més avançats, que poden oferir una experiència més fluida i realista.

ARKit utilitza les funcions de maquinari i programari de iPhone i iPad per integrar perfectament el contingut digital amb el món real. A continuació, es presenten algunes de les característiques i funcionalitats clau d'ARKit:

[1] LiDAR és una tecnología de teledetección que utiliza pulsos de luz láser para medir la distancia a un objeto o superficie

1. **Seguiment de moviment**: ARKit pot utilitzar la càmera i els sensors del dispositiu per seguir la posició i la direcció del dispositiu en l'espai tridimensional.
2. **Entendre l'entorn**: ARKit pot detectar i analitzar l'estructura geomètrica de l'entorn circumdant, com ara superfícies planes, horitzontals i verticals.
3. **Estimació de la il·luminació**: ARKit és capaç de detectar les condicions d'il·luminació de l'entorn i ajustar automàticament els efectes de llum de l'entorn.

Amb aquestes funcionalitats, ARKit proporciona als desenvolupadors eines potents per crear aplicacions de realitat augmentada riques i interactives en dispositius iOS. Els desenvolupadors poden utilitzar els llenguatges Swift[2] o Objective-C[3], combinats amb Xcode[4] i ARKit, per construir i provar les seves aplicacions d'AR.

2.1.1 Requisits funcionals

El nostre requisit funcional és desenvolupar un SDK complet per a LogMeal APP a la plataforma iOS. Aquest SDK ha de satisfer les següents funcionalitats:

1. Obtenir els fotogrames bàsics de la càmera.
2. Capturar dinàmicament X fotogrames, assegurant que cadascun d'ells estigui dins d'un rang d'angle específic.
3. Implementar l'algorisme d'inicialització del centre d'escena de la captura.
4. Proporcionar un mètode per actualitzar dinàmicament el centre d'escena.
5. Implementar l'algorisme per calcular l'angle basat en altitude i azimuth.
6. Implementar l'algorisme per calcular l'angle basat en entre el centre d'escena i la primera càmera o fotograma.
7. Investigar i implementar el millor rang d'angles per capturar les imatges de manera còmoda.
8. Crear Input de l'API del SDK, generar un fitxer JSON que contingui la informació important de la captura i la images.
9. Crear una seqüència de vídeo basada en els fotogrames específics capturats.
10. Implementar un mètode per seleccionar la imatge principal, que és la imatge RGB obtinguda quan l'angle de captura del fotograma s'apropa als 90 graus, sent la imatge de captura més rellevant.
11. Enviar totes les dades generades al servidor de l'API de LogMeal per obtenir un model d'estimació de volum i quantitat d'aliments.
12. Crear la interfície del SDK per integrar-la amb React Native.

[2]Swift és un llenguatge de programació creat per Apple per al desenvolupament d'aplicacions per a iOS.

[3]Objective-C també és un llenguatge de programació per al desenvolupament d'aplicacions per a iOS, i va ser precursor de Swift.

[4]Xcode és un IDE creat per Apple per al desenvolupament d'aplicacions per a dispositius com iOS i macOS.

2.1.2 Requisits no funcionals

Requisits no funcionals requereixen determinar com ha de funcionar l'usuari mentre captura. Aquestes necessitats s'han estudiat a fons per aconseguir el millor equilibri entre resultats i experiència d'usuari. L'objectiu principal del SDK és enviar les entrades necessàries al servidor de l'API LogMeal. Per aconseguir aquest objectiu, els usuaris han de realitzar les següents accions mentre capturen aliments específics:

1. Moure lentament i de manera controlada el dispositiu de dreta a esquerra o d'esquerra a dreta, dibuixant una trajectòria arquejada Fig. 2.1 i mantenint sempre la càmera enfocada a l'aliment. Recomanem moure's de dreta a esquerra ja que és més natural en dispositius mòbils, tot i que també és efectiu fer-ho d'esquerra a dreta.

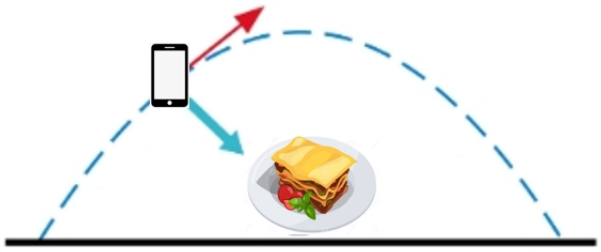

Figure 2.1: La trajectòria de moviment esperada de l'usuari quan utilitza el dispositiu, la fletxa vermella que indica que el dispositiu es mou cap a la dreta. El menjar es col·loca al centre de l'escena i la càmera s'ha d'apuntar cap al menjar tal com indica la fletxa blava.

2. Implementar una funcionalitat de correcció raonable. Si l'usuari es mou massa ràpidament o no amb prou detall, es possible que perden alguns fotogrames clau durant el procés de captura, el sistema ha de ser capaç de corregir-ho automàticament. Tot i que l'usuari no pugui capturar tots els fotogrames, només cal capturar almenys 30 fotogrames amb la imatge principal, és a dir, la que està més a prop d'un angle de 90 graus, per que la captura sigui vàlida. En cas contrari, l'usuari haurà de tornar enrere per capturar els fotogrames clau necessaris per fer una captura vàlida.

3. Es requereix un retard d'1 a 2 segons en el moment d'iniciar la captura. Després de diverses proves, s'ha observat que l'algorisme és molt inestable durant les primeres iteracions del procés de captura per calcular punt centre d'escena, per la qual cosa cal esperar diverses iteracions per obtenir el valor central de l'objecte.

4. L'usuari seguirà una trajectòria en forma d'arc, compresa entre els angles 50 i 130, abastant un total de 80 graus Fig. 2.2. Segons la recerca de *Maria Fernanda*, ha determinat que el nombre òptim de fotogrames per capturar és 50. Aquesta quantitat ofereix una representació en 3D adequada, ja que cada fotograma capturat cobreix 80 / 50 = 1.6 graus; proporcionant una visió completa. Es va decidir limitar la captura a 50 fotogrames per evitar un augment considerable en la mida del vídeo final. L'objectiu principal és optimitzar l'enviament eficient de la seqüència de vídeo a l'API per tal de minimitzar el temps d'espera de l'usuari després de finalitzar la captura fins al processament de les dades.

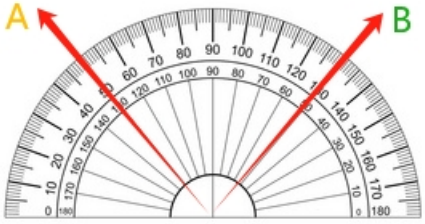

Figure 2.2: El rang de moviment de l'usuari va des de l'angle inicial A (50 graus) fins a l'angle final B (130 graus), amb un total de 80 graus.

2.1.3 Requisits d'usuari

Pel que fa als requisits de l'usuari, la nostra idea és tenir en compte els diferents costums d'ús de la mà esquerra i dreta, per la qual cosa es proporcionen diferents maneres de moviment perquè l'usuari pugui operar el dispositiu mòbil de manera més natural. En iniciar el mode de captura, el sistema mostra una interfície d'espera i ofereix un breu tutorial animat que explica detalladament com executar correctament les accions de moviment i tota la informació necessària. Ens assegurarem que la interfície d'usuari sigui fàcil d'utilitzar i proporcionarem un tutorial detallat pas a pas per a la captura. Tot i que aquesta és només una idea nostra, aquesta part ha de ser implementada en el desenvolupament de Front-end en React Native, ja que no forma part del procés de desenvolupament del SDK ni del nostre projecte. Tanmateix, hem dissenyat una interfície d'usuari senzilla per facilitar les proves pels testers, com explicarem a la secció 2.4.

Finalment, sempre tindrem en compte els escenaris d'ús de l'usuari, especialment quan es prepari per menjar i faci la captura. Ens comprometem a assegurar que l'usuari només hagi de fer una captura i no hagi de repetir el procés. Per tant, afegim totes aquestes necessitats addicionals per millorar la usabilitat del sistema.

2.2 Diagrama de flux

A continuació es mostra el diagrama de flux Fig. 2.3 que descriu el procés d'execució del programa i els diferents passos que es produeixen després que l'usuari prem el botó Iniciar gravació.

En primer lloc, en iniciar-se, el programa inicialitza totes les variables necessàries per al seu funcionament. Això inclou inicialitzar les variables del programa, cridar a la interfície de l'SDK, inicialitzar les variables de l'SDK (com ara establir l'interval d'angles per a la captura de fotogrames), utilitzar valors predefinits per inicialitzar el centre de l'escena i inicialitzar la trajectòria que conté tota la informació de la posició de la càmera detectada.

Un cop finalitzada la inicialització, l'usuari pot prémer el botó per iniciar la funcionalitat de l'SDK i començar a capturar fotogrames de la càmera. El programa primer obté la posició de la càmera del fotograma actual. Cal determinar si aquest fotograma és el primer fotograma; segons la inicialització del centre de l'escena i l'algoritme de càlcul d'angles, per al primer fotograma, el programa confirma el centre de l'escena; per al segon fotograma, el programa comença a calcular

l'angle respecte al primer fotograma. Si es manté prou temps en la posició del primer fotograma, això ajuda a calcular i actualitzar de manera efectiva els angles dels fotogrames següents respecte al centre i al primer fotograma.

A partir del segon fotograma, el programa entra en una sèrie de processos de càlcul i actualització. Primer, es calcula i s'actualitza el centre de l'escena; després, s'actualitza el pes del centre per garantir la seva precisió. Un cop determinat el centre, el programa calcula l'angle entre el fotograma actual i el primer fotograma, i determina en quin rang de la trajectòria es troba aquest angle. A continuació, es comprova si en les iteracions anteriors ja s'ha desat la informació de posició i la imatge d'aquest angle. Si no, es desa la informació actual; si ja existeix, es manté la informació més valuosa. Després de completar aquests passos, el programa processa el següent fotograma i entra en la següent iteració.

Quan s'obté prou informació de fotogrames o s'arriba al nombre necessari de fotogrames, el programa completa la captura i obté tota la informació necessària per a l'API. A continuació, es processa tota la informació i es codifica el vídeo per oferir al servidor. Després, el programa atura l'execució i espera per enviar les dades al servidor de l'API de LogMeal.

A més, l'usuari pot prémer el botó de parada en qualsevol moment. Encara que no es mostri al diagrama de flux, peró aquest botó és aplicable en qualsevol fase del programa. Quan l'usuari prem aquest botó, el programa atura l'execució i detecta i desa tota la informació obtinguda fins al moment. Si la informació és suficient i inclou la imatge RGB principal, s'envia a l'API; en cas contrari, es descarten totes les dades.

Per garantir l'estabilitat del programa i l'exactitud de les dades, en cada fotograma capturat i processat, el programa realitza diverses verificacions de dades i maneig d'errors. Si es detecta qualsevol situació anòmala, el programa registra la informació d'error corresponent i intenta recuperar o repetir el procés d'obtenció de dades. Aquestes mesures pretenen millorar la robustesa del programa i l'experiència de l'usuari.

En resum, aquest disseny de flux pretén oferir dades per a un model d'estimació de volum i quantitat d'aliments mitjançant la captura i processament precís i continu de fotogrames, així com un control flexible de l'usuari, i aconseguir una integració eficient amb l'API de LogMeal.

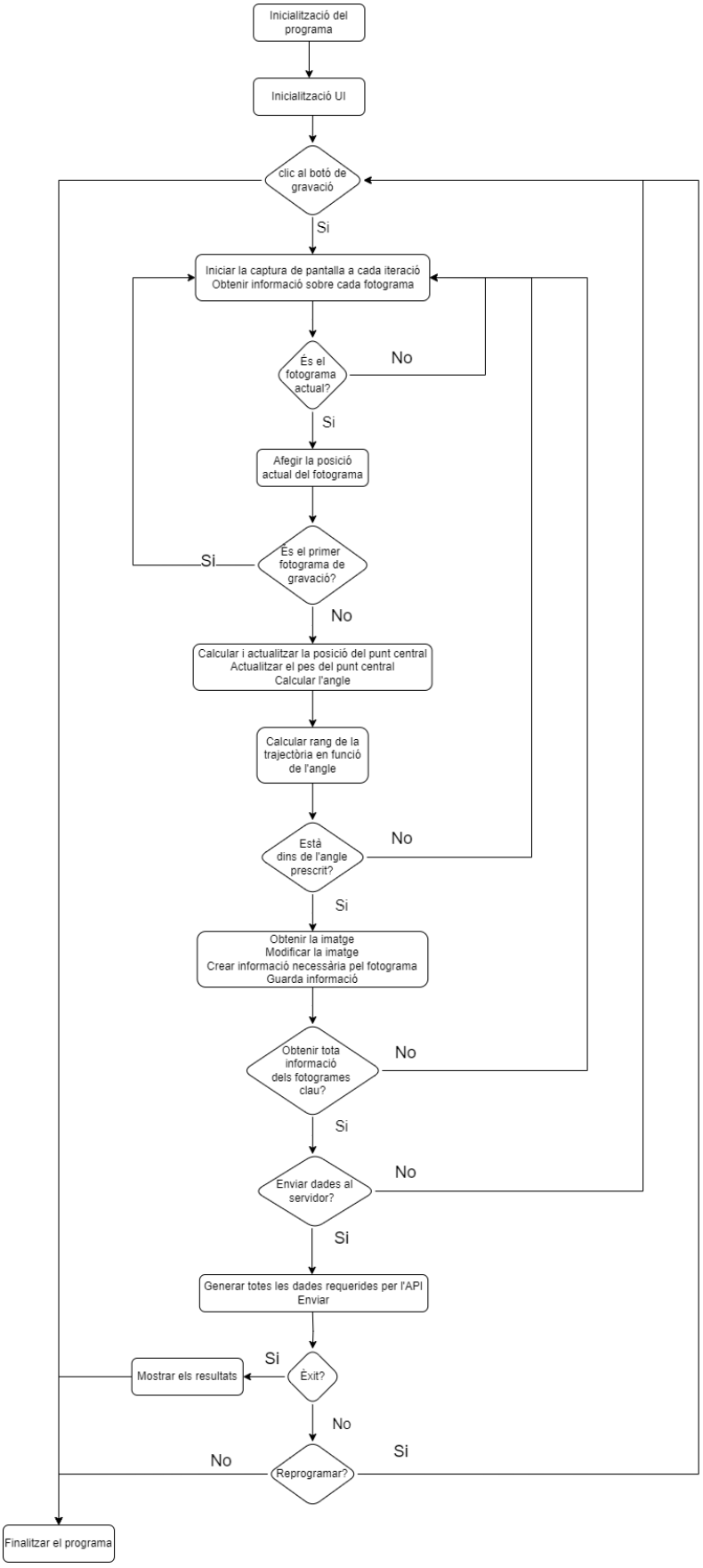

Figure 2.3: Diagrama de flux

2.3 Disseny de l'arquitectura

En aquesta secció, discutirem el diagrama d'estructura Fig. 2.4 del projecte. El diagrama següent no és l'estructura completa; només mostrarem les parts més importants, per mantenir el diagrama net i comprensible, s'ignoraran les parts menys importants, les variables i els mètodes.

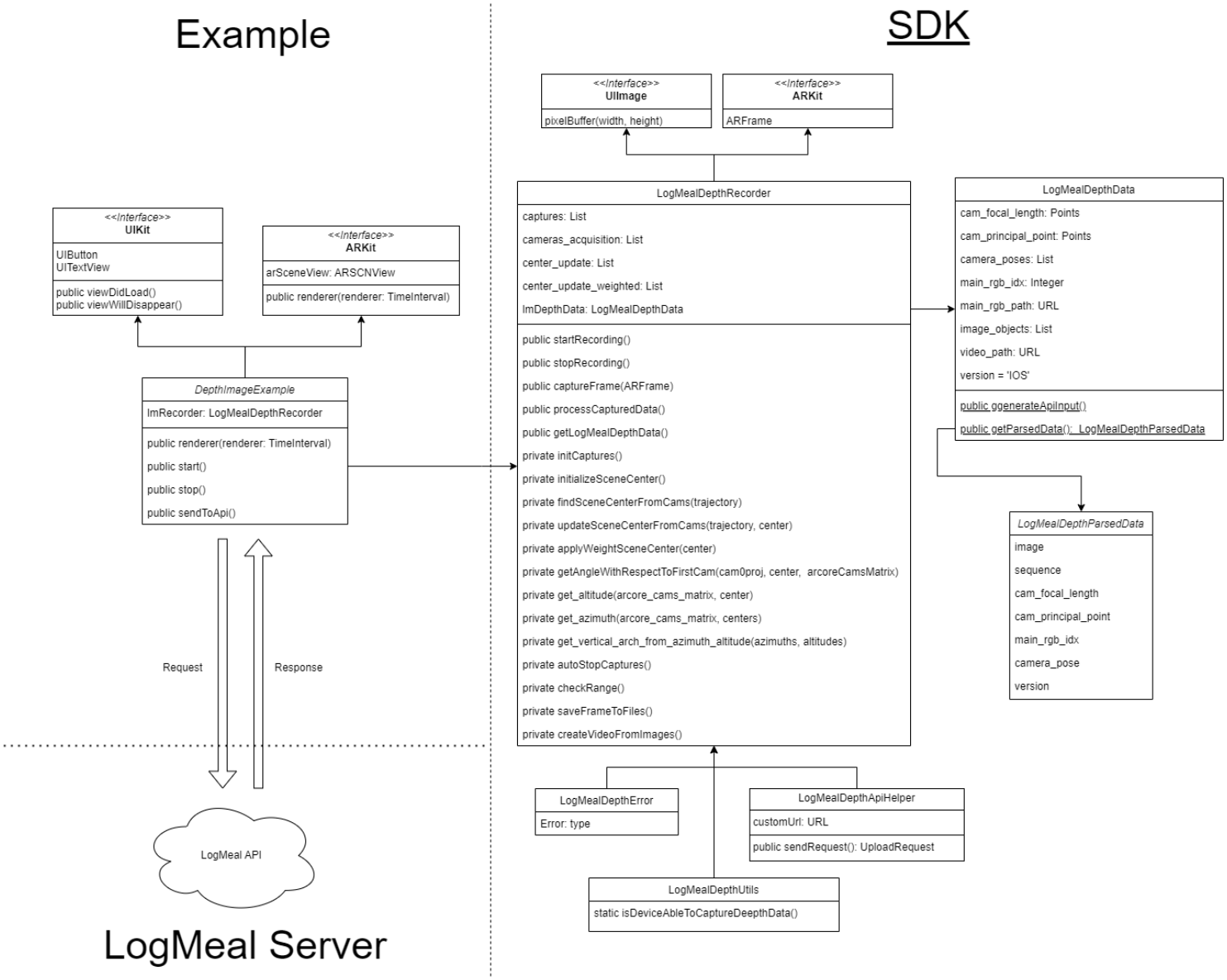

Figure 2.4: Diagrama de disseny de l'arquitectura

Com es mostra a la figura anterior, hem dividit el projecte en dues parts: Example i SDK. A la part d'Example, mostrem principalment com integrar i cridar les funcionalitats del SDK per facilitar les proves i presentar els resultats de manera visual. A l'Example, utilitzem *UIKit*[5] per construir els components d'UI necessaris, com ara botons i caixes de text. A ARKit, fem servir

[5]UIKit és framework d'interfícies d'usuari (UI) per a aplicacions iOS d'Apple

el monitor de *ARSceneView*[6] per mostrar les dades de la càmera i utilitzem un mètode de renderització d'ARKit per capturar imatges de l'entorn i obtenir informació de cada fotograma de la càmera. A la part final de secció *d'implementació obtenició de les dades del fotograma*, explicarem detalladament com utilitzar ARKit per cridar el SDK.

A la part del SDK, s'inclou el codi principal del programa. La lògica de funcionament del SDK ja s'ha descrit detalladament al diagrama de flux, per la qual cosa no repetirem aquí. A l'SDK, la classe principal és *LogMealDepthRecorder*, que sobrecargar un mètode de *UIImage*[7] per processar les imatges obtingudes, i utilitza els atributs de *ARFrame*[8] de ARKit per representar un fotograma capturat per la càmera en un moment específic i les dades de seguiment associades. Utilitzem aquestes dades per calcular el punt central i l'angle necessaris. A *LogMealDepthRecorder*, implementem diversos algoritmes i variables per calcular el punt central i l'angle, que es detallaran a la secció d'implementació. La classe *LogMealDepthRecorder* utilitza *LogMealDepthData* per abstraure les dades obtingudes i emmagatzemar-les a la memòria cache del dispositiu, per generar millor les dades necessàries per a l'API de LogMeal. Potser també has notat la presència de *LogMealDepthParseData*, que, similar a *LogMealDepthData*, s'utilitza per abstraure i emmagatzemar les dades. La diferència és que *LogMealDepthData* s'utilitza per a la informació de cada fotograma, mentre que *LogMealDepthParseData* processa les dades per generar les necessàries per a l'API. A més, també tenim algunes classes d'eines per executar les funcions del SDK de manera més eficient.

2.4 Interfície d'usuari

En aquesta secció, explicarem interfície d'usuari d'exemple utilitzada per provar l'SDK, així com les millores fetes en base al feedback dels clients per sigui més adequada als hàbits dels usuaris.

En les tres imatges següents Fig. 2.8, es mostren tres estats diferents d'interfície d'usuari d'exemple. De dreta a esquerra, es troba l'estat en què el programa s'inicia i es prepara per començar a capturar imatges Fig. 2.5. Al centre de la imatge, es veu clarament l'objecte capturat, que també és el que volem per fer a model d'estimació de volum i quantitat d'aliments. Mitjançant la imatge central i els serveis proporcionats per ARKit, podem obtenir fàcilment les dades de fotogrames necessàries. A sobre del imatge capturat, podem configurar l'API del servidor per facilitar la depuració durant el procés de desenvolupament. I a sota de la imatge capturat, disposem de tres botons: **Start**, **Send to API** i **Stop**. Les funcions d'aquests botons corresponen als tres mètodes de l'exemple i al flux del diagrama. A la part inferior de la zona de botons, hi ha una barra d'estat d'informació que ens permet comprendre clarament l'estat de l'execució del programa, els resultats o els errors.

A la segona imatge Fig. 2.6, quan iniciem el programa i capturem el vídeo, la barra d'estat d'informació ens recorda que s'està duent a terme una tasca. Durant aquest temps, el botó **Stop** és accessible, el que significa que podem aturar o finalitzar anticipadament la captura de vídeo.

[6] ARSceneView és una classe de ARKit que permet mostrar objectes de realitat augmentada en dispositius iOS a través de la càmera del dispositiu.

[7] UIImage és una classe de UIKit en iOS que representa una imatge.

[8] ARFrame és una classe en ARKit que representa un fotograma capturado.

Figure 2.5: Estat inicial

Figure 2.6: Estat procés

Figure 2.7: Estat final

Figure 2.8: UI per a provar l'SDK

A la darrera imatge 2.7, un cop hem obtingut totes les dades de fotogrames i imatges necessàries, podem generar les dades de l'API necessàries i enviar-les al servidor mitjançant el botó **Send to API**. Quan el servidor les processa amb èxit, també rebem una confirmació corresponent.

Figure 2.9: Example de Magic Scan. Font Magic Scan. 2022. Magiscan Inc.

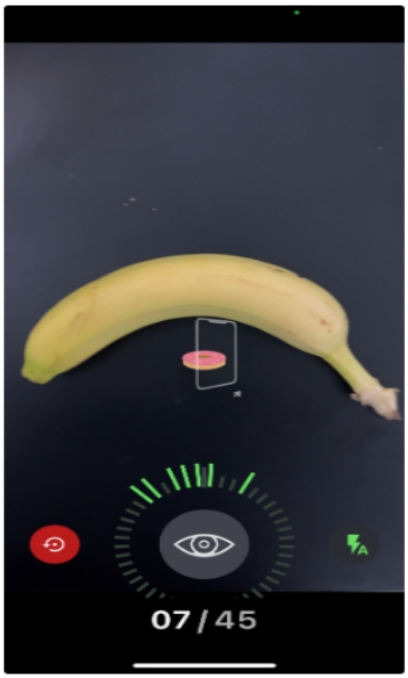

Figure 2.10: Example de UI ideal

Figure 2.11: UI ideal per l'usuaris

A continuació, demostrarem les nostres idees per al disseny de la interfície d'usuari de l'aplicació 2.11, amb l'objectiu de garantir que l'usuari sempre pugui conèixer l'estat actual de la captura.

Tenim la intenció d'utilitzar una interfície circular similar a la de l'aplicació *MagiScan* [9] 2.9, però no serà completament circular de 360 graus, sinó que només inclourà un arc de semicercle entre els 50 i els 130 graus, com es mostra a la figura Fig. 2.2 de la secció 2.1.2. Hem triat aquest rang d'angles perquè s'adequa millor als hàbits d'ús dels usuaris, ja que no és necessari capturar de manera completa els 360 graus, i els 80 graus d'angle és més adequats per a la rotació de la munyeca. Gràcies als serveis que ofereix Logmeal API, quan només tenim 80 angles, també podem mostrar als usuaris un model d'estimació de volum i quantitat d'aliments. A més, tenim previst utilitzar els colors per alertar els usuaris quan no capturin o perdin cert rang d'angles, mostrant la interfície en color vermell en aquests casos 2.10.

D'altra banda, considerem que és crucial mostrar un compte enrere d'uns 2 segons a la interfície quan l'usuari comenci la captura. Això ajudarà els usuaris a aconseguir un punt central més estable, optimitzant el procés de captura. Per tant, la interfície inclourà una petita animació de compte enrere en iniciar-se, juntament amb instruccions animades sobre com realitzar una captura efectiva.

Un cop completada la captura, l'usuari serà redirigit a una altra interfície, on podrà veure les dades dels aliments que acaba de capturar. Com que el processament de les dades trigar entre 8 i 10 segons al servidor, serà necessari mostrar un missatge de progrés de processament perquè l'usuari pugui veure els resultats finals quan s'hagi completat el procés de processament.

Finalment, cal destacar que el disseny de la nostra interfície també inclourà una funció de alertar els usuaris si estan actuant massa ràpid, o si es perd algun fotograma clau i per notificar dinàmicament quan la captura ha assolit el valor mínim efectiu requerit per a l'API. Quan el programa detecti que s'han capturat tots els fotogrames clau necessaris, es notificarà a l'usuari que ja s'ha aconseguit el requisit mínim per enviar la informació a l'API, de manera que pugui estalviar temps de captura i d'espera.

Tal com vam comentar a la secció 2.1.3, la part ideal de la UI 2.11 s'ha de desenvolupar a la interfície frontal en React Native, i no forma part del nostre procés de desenvolupament del SDK ni del nostre projecte. Per tant, aquí només estem mostrant l'efecte complet que volem que tingui el programa final. Tot i això, hem dissenyat una interfície d'usuari simple 2.8 per facilitar les proves als testers, que inclou totes les funcions necessàries.

2.5 Implementació

En aquest capítol, discutirem detalladament la implementació específica del SDK, dividint la funcionalitat en cinc mòduls. Primer, obtindrem les dades del fotograma capturat actual, incloent-hi l'anàlisi de les dades específiques que necessitem, com la informació de la imatge i la postura de la càmera. Un cop obtingudes les dades del fotograma capturat, podrem començar a calcular el punt central d'escena. Aquest pas és important, perque és la base per al càlcul de l'angle posterior. A continuació, presentarem dos mètodes per calcular l'angle i triarem l'algoritme més

[9] MàgiScan és una eina que utilitza la realitat augmentada (AR) per ajudar en la detecció i seguiment de patrons de moviment.

òptim i eficient. Després de calcular cada angle, generarem el fitxer de dades del fotograma corresponent i el desarem com a fitxer temporal a la memòria cache del dispositiu, per poden generar finalment el fitxer necessari per a l'API i enviar-lo a servidor de LogMeal. Mitjançant aquests passos, hem de assegurar que la implementació del SDK sigui funcional i eficient.

2.5.1 Obtenció de les dades del fotograma

Per obtenir les dades del fotograma actual, fem servir l'objecte *ARFrame* de ARKit. L'objecte *ARFrame* de ARKit conté diverses dades per representar el fotograma d'actual, com ara imatges, estats del dispositiu, informació de detecció de planes, etc. Cal tenir en compte que, per obtenir les dades de *ARFrame* d'ARKit, el dispositiu ha de ser compatible amb la funcionalitat d'ARKit. Els models compatibles es mostren detalladament al capítol 2.3. Per utilizar l'SDK, és necessari seguir els següents passos.

En primer lloc, abans de cridar l'SDK, els desenvolupadors han d'utilitzar la classe *ARSCNView* del framework de ARKit. Necessitem l'objecte *ARSession*[10] per gestionar la sessió de seguiment d'AR. En segon lloc, cal heretar *ARSCNViewDelegate*[11] i sobrecargar el mètode *renderer(_ renderer: SCNSceneRenderer, updateAtTime time: TimeInterval)*. El mètode *renderer* és un mètode de delegat que s'usa per obtenir el contingut de l'escena en cada fotograma. En concret, aquest mètode s'executa en cada fotograma renderitzat, oferint l'oportunitat de realitzar les operacions d'actualització necessàries. Això garanteix que el contingut de l'escena estigui actualitzat en cada fotograma i es pugui ajustar segons la nova informació proporcionada per ARKit.

Aquí teniu un exemple senzill que utilitza el mètode *renderer()* per Obtenir la dades de cada fotograme:

Listing 2.1: DepthImageExampleController

```
class DepthImageExampleController: UIViewController, ARSCNViewDelegate{

    var arSceneView: ARSCNView
    var lmRecorder = LogMealDepth.SDK()

    override func viewDidLoad() {
        arSceneView.delegate = self
        if (LogMealDepthUtils.isDeviceAbleToCaptureDeepthData()) {
            let configuration = ARWorldTrackingConfiguration()
            arSceneView.session.run(configuration)
        }
    }

    func renderer(_ renderer: SCNSceneRenderer, updateAtTime time: TimeInterval) {
        guard LogMealDepthUtils.isDeviceAbleToCaptureDeepthData() else { return }

        await self.lmRecorder.captureFrame(arSceneViewCurrentFrame:
            self.arSceneView.session.currentFrame)
    }
```

[10] ARSession és una classe permet establir una connexió entre el món real capturat per la càmera del dispositiu i els objectes virtuals

[11] ARSCNViewDelegate és una interfície de delegació utilitzada en ARKit per gestionar i respondre a esdeveniments relacionats amb la vista de l'escena de realitat augmentada (ARSCNView)

```
}
```

En aquest exemple, el mètode *renderer()* corra en cada fotograma per actualitzar la posició de tots els nodes de l'escena, de manera que es mantinguin sincronitzats amb la posició de la càmera. L'objecte *self.arSceneView.session.currentFrame* proporciona totes les dades necessàries per al fotograma actual.

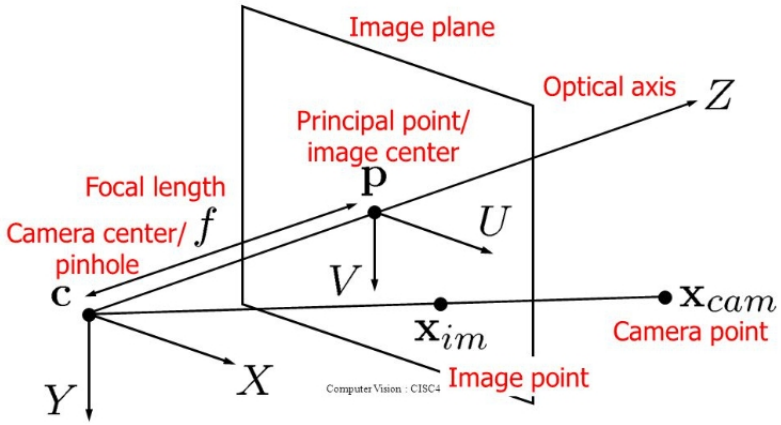

Figure 2.12: Imatge que mostra paràmetres principals de la càmera. Font: Zeng, 2017, Scene Representation.

Un cop teinm de totes les dades necessàries de la fotografia actual, hem de determinar quines d'aquestes dades Fig. 2.12 són útils per a nosaltres. En primer lloc, cal obtenir la mida del pla de la imatge i les dades RGB d'imatge. A més, necessitem el valor de la longitud focal i el punt principal. La longitud focal és un paràmetre clau del sistema òptic de la càmera, que representa la distància des de la lent fins al punt focal. Quan la longitud focal és més gran, el camp de visió és més petit, i els objectes a la imatge semblen més grans. La longitud focal es sol utilitzar en la transformació perspectiva i la correcció d'imatges de la càmera. El punt principal és el punt central al pla d'imatge de la càmera, que coincideix amb el centre òptic. Aquest punt s'utilitza en la calibració de la càmera per determinar la posició central de la imatge. En aplicacions pràctiques, conèixer la posició del punt principal ajuda en la correcció i el tractament d'imatges. Aquests dos paràmetres són molt importants en la calibració de la càmera i la fotografia, ja que ajuden a comprendre el principi de formació d'imatges de la càmera i a realitzar el tractament i la correcció d'imatges. En segon lloc, necessitem una dada fonamental: la matriu de transformació de la càmera(posició de la càmera o fotograma). Aquesta matriu s'utilitza per calcular la posició de la càmera en coordenades mundials, així com per calcular la transformació del punt central i l'angle.

Podem utilitzar la propietat *currentFrame.camera.transform* de ARKit per obtenir la matriu de transformació de la càmera Fig. 2.13. Aquesta matriu descriu la posició i la direcció de la càmera en el sistema de coordenades del món. En ARKit, la càmera és un node especial que representa la posició i la direcció del dispositiu en el món real.

Normalment, aquesta matriu de transformació és una matriu de 4x4 que descriu la transformació de la càmera des del sistema de coordenades del món fins al propi sistema de coordenades de

la càmera. Inclou informació sobre translació, rotació i escalat. Ens interessen principalment les tres primeres columnes de la matriu, que representen les direccions dels eixos X, Y i Z de la càmera.

Aquesta matriu de transformació de 4x4 té normalment la forma següent:

$$\begin{bmatrix} right_x & up_x & forward_x & position_x \\ right_y & up_y & forward_y & position_y \\ right_z & up_z & forward_z & position_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 2.13: Matriu de transformació de la càmera.

On m_{ij} representa l'element de la fila i la columna j de la matriu. En ARKit, aquesta matriu de transformació descriu la transformació de la càmera des del sistema de coordenades del món fins al sistema de coordenades de la pròpia càmera, és a dir, la posició de la càmera.

Les tres primeres columnes de la matriu de transformació (m_{11} a m_{13}) solen representar les direccions dels eixos X, Y i Z de la càmera. La darrera columna (m_{14} a m_{34}) sol representar la posició de la càmera en el sistema de coordenades del món. L'última fila és fixa, normalment [0, 0, 0, 1], per garantir la correcció de la matriu.

Mitjançant l'extracció d'aquesta informació de la matriu de transformació, podem obtenir la posició i la direcció de la càmera en el sistema de coordenades del món, permetent-nos realitzar operacions relacionades amb la càmera.

Aquí hi ha una figura Fig. 2.14 que descriu clarament la transformació de les coordenades de la càmera des del sistema de coordenades del món fins al propi sistema de coordenades de la càmera.

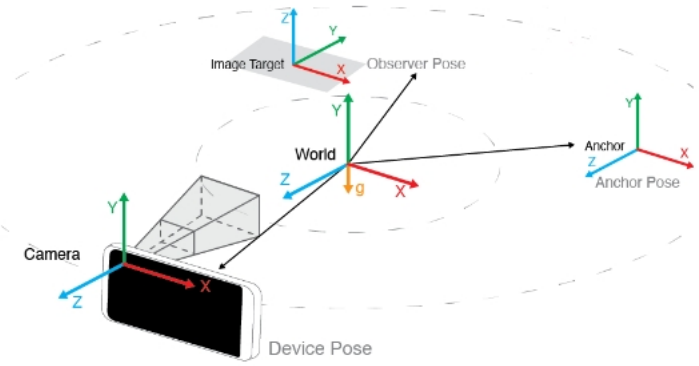

Figure 2.14: Visualització d'un dispositiu que apunta a un objecte del món real i com es relaciona amb Anchor. Font: Vuforia, 2023, Spatial Frame of Reference for Unit.

2.5.2 Computació de centre de l'escena

Per determinar l'angle relatiu de la càmera respecte al centre de l'escena, és molt important obtenir el centre de l'escena. Per a cada fotograma de la càmera, podem definir un eix òptic que passi pel centre òptic i pel punt principal Fig. 2.15 Això es coneix com a eix òptic, i es pot definir a partir de la tercera i segona columna de la matriu de la càmera.

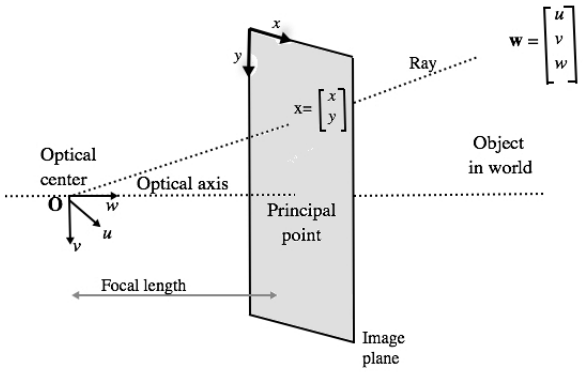

Figure 2.15: Centre òptic, l'eix òptic, la distància focal, el punt principal i el pla que defineix la imatge. Font:Azarcoya-Cabiedes, 2014, Centre òptic.

El centre de l'escena s'obté a partir del punt de creuament de totes aquestes línies, amb una mitjana ponderada Fig. 2.16, on si dues línies són paral·leles, la seva ponderació és zero. Comparant parelles de línies (linies obtingudes de dos fotogrames), podem trobar el punt de creuament de l'eix òptic (tot i que trobar un punt de creuament perfecte pot ser poc probable, busquem el punt més proper).

La següent figura Fig. 2.16 mostra la projecció de tots els eixos òptics i com es converteixen en una regió comuna. Aquesta figura és un bon exemple de com visualitzar el centre de l'escena durant la captura d'imatges.

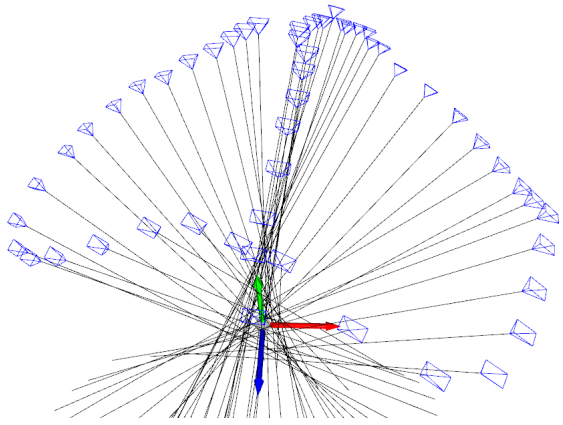

Figure 2.16: Simulacó mostra la convergència cap al centre de l'escena dels diferents camera pose obtinguts en una captura.

Dividim el càlcul del centre de l'escena en tres passos. En primer lloc, inicialitzem el punt central al començament del programa. En segon lloc, en les iteracions posteriors, ajustem el càlcul del centre. Finalment, actualitzem el pes del centre de l'escena per obtenir el punt central òptim de l'escena.

Inicialitzer el punt central

Quant a la inicialització del càlcul del centre de l'escena, fem servir la inicialització per defecte de dos elements:

1. **totw**: representa el pes del centre de l'escena, inicialitzat a 0.0.
2. **totp**: representa la posició del centre de l'escena, inicialitzat a [0.0, 0.0, 0.0], és a dir, l'origen de l'espai tridimensional."

Actualitzar el centre de l'escena des de les càmeres

Cada vegada que obtenim un nou fotograma, cridem a la funció d'actualització del centre. La funció té com a objectiu comparar cada nou conjunt de càmeres amb tots els conjunts de càmeres anteriors i actualitzar el centre i els pesos corresponents en conseqüència. A continuació, presentaré el pseudocodi del mètode d'actualització del centre de l'escena:

Algorithm 1 Update Scene Center From Cameras

```
1: function UpdateSceneCenterFromCams(trajectory : [float4x4], center : (Float, [Float]))
2:     totw, totp ← center
3:     if trajectory.isEmpty then
4:         return (totw, totp)
5:     end if
6:     lastCam ← trajectory.last
7:     lcCol3 ← lastCam.columns.3
8:     lcCol2 ← lastCam.columns.2
9:     for cam ← trajectory do
10:        tcCol3 ← cam.columns.3
11:        tcCol2 ← cam.columns.2
12:        lcCol3Array ← (lcCol3.x, lcCol3.y, lcCol3.z)
13:        lcCol2Array ← (lcCol2.x, lcCol2.y, lcCol2.z)
14:        tcCol3Array ← (tcCol3.x, tcCol3.y, tcCol3.z)
15:        tcCol2Array ← (tcCol2.x, tcCol2.y, tcCol2.z)
16:        (p, w2) ← closestPointTwoLines(oa : lcCol3Array, da : lcCol2Array, ob :
   tcCol3Array, db : tcCol2Array)
17:        if w2 > 0.00001 then
18:            totp ← totp.enumerated().map{(index, value) → value + (p[index] × w2)}
19:            totw ← totw + w2
20:        end if
21:    end for
22:    return (totw, totp)
23: end function
```

Els paràmetres d'aquest algorisme són una trajectòria que conté una sèrie de matrius de transformació de postura de càmera fins actual i l'estructura del centre de l'escena actual. El punt central (center) és una tupla de la posició i els pesos de l'escena. L'objectiu d'aquest algorisme és actualitzar el centre de l'escena basant-se en la trajectòria de la càmera i el centre actual.

Primer, la funció comprovarà si s'ha proporcionat almenys una matriu de transformació de càmera; si no n'hi ha, retornarà directament la informació del centre actual.

A continuació, expliquem el pseudocodi:

1. Obtenim la informació de posició de la darrera càmera extreient la quarta columna de la darrera matriu de transformació de la càmera (normalment representa la posició de la càmera).
    - *lcCol3* representa la quarta columna de l'última matriu de transformació de la càmera, que normalment representa la posició de la càmera (coordenades x, y, z).
    - *lcCol2* representa la tercera columna de l'última matriu de transformació de la càmera, que normalment representa la direcció o orientació de la càmera.
2. Recorrerà tota la trajectòria de la càmera i, a cada iteració, calcularà els punts més propers entre la posició de la darrera càmera i la posició actual de la càmera.
    - *tcCol3* representa la quarta columna de la matriu de transformació de la càmera actual, que també representa la posició de la càmera.
    - *tcCol2* representa la tercera columna de la matriu de transformació de la càmera actual, que també representa la direcció o orientació de la càmera.
3. Utilitzem una funció anomenada **closestPointTwoLines** per calcular els punts més propers entre dues línies i retornar el valor del pes ($w2$). Si el valor del pes és superior a un llindar (aquí establert com a 0.00001), actualitzem la posició i el pes del centre.
4. Actualitza el centre, la funció recorrerà cada component de la posició del centre, multiplicarà el punt més proper actual pel valor del pes i l'afegirà al centre original, i acumularà el valor del pes a la suma ponderada total.
5. Finalment, la funció retornarà la posició del centre actualitzada i la suma ponderada total. Aquesta fonció té com a objectiu ajustar dinàmicament la posició del centre de l'escena segons la trajectòria de la càmera per adaptar-se als moviments i canvis dels objectes de l'escena.

En forma senzil, que aquest pseudocodi té com a funció ajustar dinàmicament la posició del centre de l'escena segons la trajectòria de la càmera. Compara les noves posicions de la càmera amb les anteriors, calcula el punt més proper i actualitza la posició i el pes del centre. Això s'aconsegueix perquè el centre de l'escena es mogui amb la càmera i s'adapti als canvis i moviments dels objectes de l'escena.

Cal notar que la funció d'actualització del centre depèn completament de l'algorisme per trobar el punt més proper entre dues línies, per tant és necessari analitzar-ne el funcionament. A continuació, mostrarem el pseudocodi de l'algorisme per trobar el punt més proper entre dues línies i explicarem detalladament el seu funcionament.

Punt més proper entre dues línies

El principal principi matemàtic d'aquest algoritme implica l'ús de vectors i productes vectorials per calcular la posició del punt més proper entre dues rectes. Mitjançant càlculs vectors adequats i la selecció apropiada del paràmetre "t", es pot localitzar amb precisió el punt més proper.

En primer lloc, l'algoritme normalitza els vectors direcció per obtenir vectors unitaris. Després, calcula el producte vectorial d'aquests dos vectors unitaris per obtenir un vector perpendicular a ambdues rectes. La longitud d'aquest vector, al quadrat, proporciona una constant per calcular el punt més proper final.

A continuació, l'algoritme calcula el vector que connecta qualsevol punt a les dues rectes i utilitza la relació entre el producte vectorial i el producte escalar per calcular el paràmetre "t" que determina la posició del punt més proper entre les dues rectes.

Finalment, substituint els valors de "t" en la fórmula adequada, es pot calcular la posició exacta del punt més proper i retornar-la. Aquest procés no consisteix en una sola fórmula, sinó en una sèrie de passos basats en relacions geomètriques i càlculs vectorials.

A continuació, es mostren les principals fórmules de càlcul de la posició del punt més proper entre dues rectes:

1. En primer lloc, cal calcular el producte vectorial de les direccions de les dues rectes.

$$c = \hat{da} \times \hat{db}$$

2. A continuació, es calcula el quadrat de la longitud del vector del producte vectorial.

$$denom = ||c||^2$$

3. Després, es calcula el vector "t" que connecta qualsevol punt a les dues rectes.

$$t = ob - oa$$

4. A partir d'aquest vector "t", es calculen els paràmetres "ta" i "tb" que determinen la posició del punt més proper entre les dues rectes.

$$ta = \frac{t \cdot (db \times c)}{denom + 1 \times 10^{-10}}$$

$$tb = \frac{t \cdot (da \times c)}{denom + 1 \times 10^{-10}}$$

5. Finalment, es calcula la posició del punt més proper utilitzant els paràmetres "ta" i "tb".

$$nearest\_point = \frac{(oa + ta \times da) + (ob + tb \times db)}{2}$$

La següent és la implementació de l'algoritme en codi, que calcula la posició del punt més proper i la distància entre dues rectes tridimensionals. Accepta dues parelles de vectors com a entrada: "*oa*" i "*da*" representen un punt i un vector de direcció de la primera recta, mentre que "*ob*" i "*db*" representen un punt i un vector de direcció de la segona recta.

Algorithm 2 Closest Point Between Two Lines

```
1: function ClosestPointTwoLines(oa : float3, da : float3, ob : float3, db : float3)
2:     c ← d̂a × d̂b
3:     denom ← length(c) × length(c)
4:     t ← ob − oa
5:     ta ← (t · (db × c))/(denom + 1e − 10)
6:     tb ← (t · (da × c))/(denom + 1e − 10)
7:     if ta > 0 then
8:         ta ← 0
9:     end if
10:    if tb > 0 then
11:        tb ← 0
12:    end if
13:    return (float3((oa + ta × da + ob + tb × db) × 0.5), denom)
14: end function
```

Aquest algoritme, basat en càlculs vectors, normalitza primer els vectors direcció de les dues rectes i després calcula el producte vectorial d'aquests dos vectors per obtenir un vector perpendicular a ambdues rectes. Després calcula el quadrat de la longitud d'aquest vector. Seguidament, calcula el vector que connecta un punt de la primera recta amb un punt de la segona recta i utilitza aquest vector i el producte vectorial de les dues rectes per calcular el paràmetre "t" que determina la posició del punt més proper entre les dues rectes. Finalment, utilitza aquests paràmetres per calcular la posició del punt més proper i retorna aquesta posició, així com la distància entre les dues rectes (denominador).

Aplicar pes al centre de l'escena

Quan l'algoritme ha acabat amb la part relacionada amb el centre de l'escena, recalcularem els pesos del punt central. Aquest pas essencialment calcula la mitjana ponderada del vector de posició, on cada component es divideix pel pes total. La nostra funció retorna el punt central amb els pesos aplicats. Aquest centre s'utilitza per calcular els angles. Farem servir el següent pseudocodi per processar la mitjana ponderada dels punts centrals de l'escena després de cada fotograma, assegurant-nos que la suma dels pesos sigui més gran que zero per normalitzar-la i obtenir el punt central de l'escena actual del fotograma.

Algorithm 3 Apply Weight to Scene Center

```
1: function ApplyWeightSceneCenter(center : (Float, [Float]))
2:     totw, totp ← center
3:     if totw > 0.0 then
4:         totp ← totp.map{x → x/totw}
5:     end if
6:     return totp
7: end function
```

2.5.3 Computació de l'angle

Un cop s'ha determinat el centre de l'escena, assumim que es troba en un pla. Després, utilitzem aquest pla i la posició de la càmera per calcular diferents angles, els quals descriuen la posició de la càmera respecte al centre. Per aconseguir-ho, necessitem actualitzar el centre de l'escena i l'angle de cada càmera després de cada fotograma. D'aquesta manera, ens assegurem que la càmera pugui capturar amb precisió totes les parts de l'escena.

Pel que fa al càlcul dels angles, utilitzem dos conjunts d'algorismes. Un es basa en les **altituds** i **azimuths** de la càmera i del centre per calcular els angles, mentre que l'altre es basa en la relació entre el fotograma actual i el primer fotograma per calcular els angles. Malgrat que finalment optem per aquest segon conjunt d'algorismes, perquè el primer pot generar salts d'angles inesperats prop de 90 graus, impedint així aconseguir els objectius desitjats, Però també donarem una breu explicació del primer conjunt d'algorismes.

Obtenir l'arc vertical des de l'azimut i l'altitud

En aquest capítol, presentem breument l'algorisme per calcular els angles basant-nos en l'altura i l'azimut dels objectius de les càmeres respecte al punt central de l'escena.

Primerament, en aquest codi definim dues funcions: *get_altitude* i *get_azimuth*, les quals s'encarreguen de calcular l'altura i l'azimut respectivament, de la càmera en relació amb el centre de l'escena.

Algorithm 4 Get Altitude

```
1:  function GetAltitude(arcore_cams_matrix, center)
2:      altitude ← []
3:      for cam in arcore_cams_matrix do
4:          cam2center_vector ← cam[: 3, 3] − center
5:          z_vector ← cam2center_vector[1]
6:          xy_vector ← √((cam2center_vector[0]^2) + (cam2center_vector[2]^2))
7:          θ ← degrees(arctan(z_vector / xy_vector))
8:          altitude.append(θ)
9:      end for
10:     return altitude
11: end function
```

La funció **get_altitude** rep dos paràmetres: *arcore_cams_matrix* i *center*. *arcore_cams_matrix* és una llista que conté matrius de càmeres, on cada matriu descriu la posició i direcció de la càmera. El *center* és el punt central de l'escena. Aquesta funció calcula primer el vector de posició de cada càmera respecte al centre de l'escena i després, mitjançant la projecció d'aquest vector sobre el pla horitzontal i la seva component vertical, calcula l'altura de la càmera. Finalment, retorna totes les alçades emmagatzemades en un array.

La funció **get_azimuth** també rep dos paràmetres: *arcore_cams_matrix* i *center*. Similar a **get_altitude**, primer calcula el vector de posició de cada càmera respecte al centre de l'escena i el projecta sobre el pla horitzontal. Llavors, utilitzant el vector de posició de la primera càmera com a referència, calcula l'azimut de cada càmera. L'azimut és l'angle horitzontal respecte a una direcció de referència (sovint el nord). Aquesta funció també retorna tots els azimuts emmagatzemats en un

array.

Algorithm 5 Get Azimuth

```
function GetAzimuth(arcore_cams_matrix, center)
    azimuth ← []
    xy_proj_vector_0 ← arcore_cams_matrix[0][: 3, 3] − center
    xy_proj_vector_0[1] ← 0
    xy_proj_vector_0 ← xy_proj_vector_0 / ‖xy_proj_vector_0‖
    for cam in arcore_cams_matrix do
        cam2center_vector ← cam[: 3, 3] − center
        xy_proj_vector ← deepcopy(cam2center_vector)
        xy_proj_vector[1] ← 0
        xy_proj_vector ← xy_proj_vector / ‖xy_proj_vector‖
        θ ← degrees(arccos(xy_proj_vector_0 · xy_proj_vector))
        determinant ← (xy_proj_vector_0[0] × xy_proj_vector[2]) − (xy_proj_vector_0[2] ×
xy_proj_vector[0])
        if determinant < 0 then
            azimuth.append(θ)
        else
            azimuth.append(360 − θ)
        end if
    end for
    return azimuth
end function
```

Quan disposem de l'azimut i l'altura de les càmeres, així com del punt central de l'escena, podem utilitzar la trigonometria per calcular la diferència d'angles entre la posició de les càmeres i el centre de l'escena Fig. 2.17.

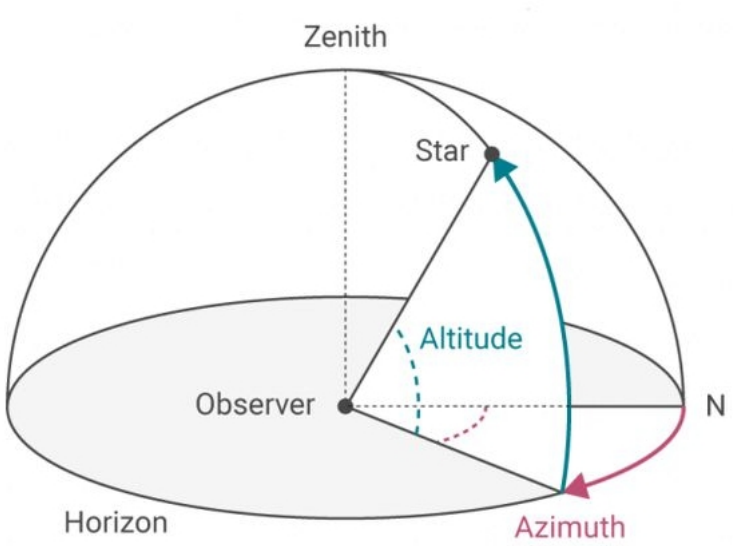

Figure 2.17: L'azimut i altitud. Font: Azarcoya-Cabiedes, 2014, The Horizontal Coordinate System

- **Azimut**: L'azimut és l'angle horitzontal de la càmera respecte a una direcció de referència (sovint el nord). Si considerem que el nord està definit com a 0 graus i girant en sentit horari fins a 360 graus, podem determinar l'angle de desviació de la càmera respecte al nord. Per exemple, si l'azimut d'una càmera és de 45 graus, la seva desviació respecte al nord serà de 45 graus.
- **Altitude**: L'altura és l'angle vertical de la càmera respecte al pla horitzontal. Normalment, definim el pla horitzontal com a 0 graus i mesuram l'angle cap amunt fins arribar a 90 graus. D'aquesta manera, podem determinar l'angle de la càmera respecte al pla horitzontal. Per exemple, si l'altura d'una càmera és de 30 graus, es troba a una posició 30 graus per sobre del pla horitzontal.

Ara, amb l'azimut i l'altura de les càmeres, podem utilitzar funcions trigonomètriques per calcular la diferència d'angles entre la posició de les càmeres i el centre de l'escena. Concretament:

- Podem utilitzar la funció sinus (sin) per calcular l'altura de la càmera respecte al pla horitzontal. El valor de la funció sinus varia entre 0 i 1, representant canvis d'angle entre 0 graus i 90 graus.
- Podem utilitzar la funció cosinus (cos) per calcular l'azimut de la càmera respecte a una direcció de referència. El valor de la funció cosinus varia entre -1 i 1, representant canvis d'angle entre 0 graus i 180 graus.

Mitjançant aquestes funcions, podem convertir l'azimut i l'altura de les càmeres en valors numèrics que indiquen la posició de la càmera respecte al pla horitzontal i a la direcció de referència. Llavors, podem utilitzar aquestes diferències entre valors numèrics per calcular la diferència d'angles entre les càmeres i el centre de l'escena.

Resumint, utilitzant l'azimut i l'altura, podem calcular la diferència d'angles entre la posició de les càmeres i el centre de l'escena mitjançant funcions trigonomètriques, cosa que ens ajuda a comprendre la posició i la direcció de les càmeres.

Algorithm 6 Get Vertical Arch From Azimuth and Altitude

1: **function** GetVerticalArchFromAzimuthAltitude(azimuths, altitudes)
2: azimuths_closest_quadrants $\leftarrow$ (azimuths $>$ 0) $\wedge$ (azimuths $\leq$ 90) $\vee$ (azimuths $\geq$ 270) $\wedge$ (azimuths $<$ 360)
3: altitudes_vertical_arch $\leftarrow$ altitudes
4: altitudes_vertical_arch[$\sim$ azimuths_closest_quadrants] $\leftarrow$ 180 $-$ altitudes_vertical_arch[$\sim$ azimuths_closest_quadrants]
5: **return** altitudes_vertical_arch
6: **end function**

La funció **get_vertical_arch_from_azimuth_altitude** rep dos paràmetres, *azimuths* i *altitudes*, que representen l'azimut i l'altura de les càmeres. La seva funció és ajustar l'altura segons l'azimut per tal que l'altura es trobi dins del rang de l'arc vertical.

Primer, filtra els azimuths per trobar aquells més propers al nord i al sud. Això s'aconsegueix

mitjançant operacions lògiques booleanes per determinar si l'azimut es troba entre 0 i 90 graus o entre 270 i 360 graus, que corresponen a les direccions nord-est i sud-oest de l'horitzó.

Després, per a aquells altitudes que no es troben en aquests rangs, ajusta l'altura mitjançant una simple relació matemàtica. En concret, substitueix aquestes altituds per 180 graus menys el valor original de l'altura. Aquesta operació fa que les altituds originalment per sota de la línia horitzontal es mapegin per sobre d'aquesta línia, creant així un arc vertical.

En resum, aquesta funció ajusta les altituds en funció dels azimuths per tal que es trobin dins del rang de l'arc vertical, permetent el càlcul de l'angle de l'arc.

No obstant això, mitjançant aquesta implementació no podem obtenir resultats numèrics completament precisos, ja que l'algorisme fa alguns salts entre els angles, el que fa impossible la captura de fotogrames. Per exemple, en aquesta implementació, una vegada es detecta un angle de 70 graus, salta directament a 100 graus. Això ens fa perdre els fotogrames corresponents als 30 graus intermedis, la qual cosa afecta greument els nostres resultats. Per tant, ens veiem obligats a centrar-nos en la recerca i implementació d'un altre algorisme.

Obtenir l'angle respecte a la primera fotograma

En el segon algorisme, utilitzem el càlcul de l'angle entre cada càmera i la primera càmera. En concret, assumeix que la posició de la càmera es defineix per la tercera columna de la matriu de la càmera i que hi ha múltiples càmeres en una escena, totes elles amb la mateixa posició central com a referència.

Algorithm 7 Get Angle with Respect to First Camera

```
function GetAngleWithRespectToFirstCam(arcore_cams_matrix, center)
    relative_theta ← []
    cam2center_vector_0 ← arcore_cams_matrix[0][: 3, 3] − center
    cam2center_vector_0 ← cam2center_vector_0 / ‖cam2center_vector_0‖
    for cam in arcore_cams_matrix do
        cam2center_vector ← cam[: 3, 3] − center
        cam2center_vector ← cam2center_vector / ‖cam2center_vector‖
        θ ← degrees(arccos(cam2center_vector_0 · cam2center_vector))
        relative_theta.append(θ)
    end for
    return relative_theta
end function
```

En primer lloc, calcula el vector des de la posició de la primera càmera fins al centre de l'escena i normalitza aquest vector per obtenir un vector unitari que representa la direcció de la càmera cap al centre de l'escena. A continuació, recorre cada matriu de càmera i calcula el vector des de la posició de cada càmera fins al centre de l'escena, normalitzant-lo també. Després, calcula l'angle entre la primera càmera i la càmera actual utilitzant el teorema del cosinus Fig. 2.18, converteix l'angle a graus i l'afegeix a una llista.

El teorema del cosinus per calcula l'angle actual respecte a la primera fotograma i el punt central:

$$\cos(\theta) = \frac{a^2 + b^2 - c^2}{2ab}$$

$$\theta = \arccos\left(\frac{a^2 + b^2 - c^2}{2ab}\right)$$

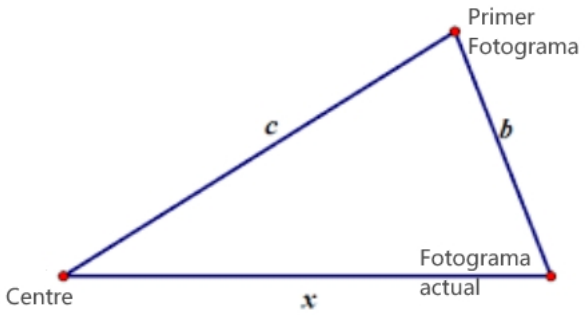

Figure 2.18: Example d'obtenir l'angle respecte a la primera fotograma.

Finalment, converteix la llista dels angles entre cada càmera i la primera càmera en un array i retorna aquest array. D'aquesta manera, en cridar aquesta funció, obtindràs un array amb els angles entre cada càmera i la primera càmera. Normalment, només necessitem l'últim d'aquests angles, ja que representa l'angle entre el fotograma actual i el primer.

2.5.4 Generació dels fitxers

En aquesta secció, explicarem com generar els nostres fitxers de resultat. Primer, hem de generar la informació de cada **fotograma**, incloent la informació de la imatge i la posició, i desar la imatge mateixa. A continuació, hem de crear un fitxer **apiInput.json** que contingui totes les dades necessàries per al servidor API. També necessitem una **imatge principal** per mostrar la foto a 90 graus. Finalment, generarem un **vídeo** que combini totes les imatges de cada fotograma. Un cop tinguem aquests quatre fitxers, podrem enviar-los al servidor API.

Guarda imatges i informació de cada fotograma

Per a cada fotograma, necessitem conservar la següent informació: les dimensions de la imatge (**width i height**), la distància focal (**focallength**) en els punts x i y, els punts principals **(principal points**) en els punts x i y, la matriu de posició (**model matrix**) per a aquest fotograma, i desar la **imatge** mateixa. A continuació es mostra un exemple:

```
imageRegionCoordinates-0:0
imageRegionCoordinates-1:0
imageRegionCoordinates-2:0
```

```
imageRegionCoordinates-3:0
width:800.0
height:600.0
fx:530.3579
fy:530.3579
cx:316.6366
cy:241.7303
modelMatrix:
:-0.915133
:-0.31407762
:0.2527583
:0.0
:0.25357568
:0.038980648
:0.9665298
:0.0
:-0.3134181
:0.9485967
:0.04396997
:0.0
:-0.13391627
:0.27718368
:-0.16490367
:1.0
```

- f_{xy} indica la distància focal en els punts x i y.
- c_{xy} indica els punts principals en els punts x i y.
- *modelMatrix* indica la informació de la matriu de posició per a aquest fotograma.
- *width* i *height* indiquen les dimensions de la imatge, aquesta informació ens ajudarà a calcular el punt central de la imatge.

Arxiu JSON (apiInput.json)

Només tenir la informació i les imatges de cada fotograma no és suficient; necessitem la posició i les imatges de tota la trajectòria. A continuació es mostra un exemple en què utilitzem aquest fitxer JSON per representar totes les posicions i informació de l'escaneig de la càmera. El més important és **camera_poses**, que conté la informació de la matriu de posició de cada fotograma clau. A més, necessitem marcar quina imatge és la nostra imatge principal(**main_rgb_idx**). En aquest cas, es marca la imatge número 24. Això significa que és la 24a imatge que vam guardar abans, I necessitem el **cam_focal_length** i el **cam_principal_point** per representar la distància focal i el punt principal finals, i que són les dades generades per l'apartat anterior de 24a informació de fotograma. A més, hem de confirmar que utilitzem la **versió iOS**, de manera que el servidor pugui processar els resultats de manera diferent segons la versió de sistema.

```
{
  "cam_focal_length" : {
    "x" : 530.35791015625,
    "y" : 530.35791015625
  },
  "cam_principal_point" : {
    "x" : 316.6365966796875,
    "y" : 241.73030090332031
  },
  "camera_poses" : {
    "0000" : [
      -0.95170885324478149,
      -0.17330189049243927,
      0.25340977311134338,
      0,
      0.28114929795265198,
      -0.82353806495666504,
      0.49268656969070435,
      0,
      0.12330909073352814,
      0.54014021158218384,
      0.83249223232269287,
      0,
      -0.032827794551849365,
      0.13469815254211426,
      -0.026005730032920837,
      0.99999994039535522
    ],
    "0001" : [
        ...
    ],
    ...
  },
  "main_rgb_idx" : 24,
  "version" : "ios"
}
```


Imatge principal

Com s'hem comentat anteriorment, la imatge principal és indispensable per al funcionament del programa, per la qual cosa és vital assegurar-se que hi ha una imatge principal adequada. La foto principal s'hauria de seleccionar de la més propera possible als 90 graus, ja que utilitzar una imatge presa des de dalt cap avall permet que els algorismes del servidor reconeguin millor la imatge i determinin el punt central del model.

Seqüència de Vídeo

A més, també necessitem generar una seqüència de vídeos perquè el servidor pugui reconèixer millor els objectes. En el SDK, fem servir un algorisme senzill de concatenació de vídeos per unir totes les imatges recopilades anteriorment.

Segons els requisits de l'API, el vídeo ha de seguir el següent format:

- Resolució: 480 x 360 píxel
- Durada de cada fotograma: 60 frame / segons
- Format del vídeo: AVI

Cal tenir en compte que, degut al pes del vídeo generat, pot afectar l'eficiència de la càrrega al servidor, per la qual cosa hem de codificar una part de les imatges de la seqüència de vídeos per reduir el seu tamany. A més, iOS només admet els formats de vídeo MOV i MP4, mentre que el nostre servidor només admet el format AVI, per la qual cosa necessitem realitzar una sèrie de passos de transcodificació abans de pujar-lo al servidor.

2.5.5 Crida a l'API

Quan tenim totes les dades necessàries, incloent apiInput.json, la seqüència de vídeo, la imatge principal del fotograma i la informació principal del fotograma, podem enviar-les al servidor a través del port que ofereix l'API de LogMeal. Cal destacar que necessitem proporcionar un **token d'autenticació**[12] per poder enviar les sol·licituds, sinó no serà permès.

Un cop tenim totes les dades necessàries, podem fer servir una sol·licitud **POST HTTPS**[13] i emprar el format **multipart/form-data**[14] per portar totes les dades necessàries i començar a enviar-les.

Després d'enviar-ho amb èxit, caldrà esperar 8-10 segons perquè el servidor processi la nostra sol·licitud i ens retorni una resposta. Quan rebem una resposta amb un codi d'estat entre 200 i 299, significa que la nostra sol·licitud ha tingut èxit; en cas contrari, ha fallat. Per exemple, el codi 400 indica una sol·licitud incorrecta, potser a causa de dades que no compleixen els requisits del servidor; el 401 indica que no està autoritzat, pot ser a causa de la falta de token d'autenticació o per altres motius que impedeixin la sol·licitud; finalment, el 500 indica un problema al servidor.

[12]Un token d'autenticació és una clau o una cadena de caràcters que s'utilitza per verificar la identitat d'un usuari durant el procés d'autenticació.

[13]POST HTTPS fa referència a una petició HTTP realitzada a través del protocol HTTPS utilitzant el mètode POST. POST: És un dels mètodes utilitzats en el protocol HTTP per enviar dades a un servidor per a processament.

[14]multipart/form-data és un tipus de codificació de dades utilitzat en les peticions HTTP. És comúment utilitzat quan s'envien contenen arxius adjunts.

Aqui tenim un exemple que mostrar com enviar sol·licituds a API de LogMeal. Hi ha diverses maneres de fer-ho, però simplement farem una demostració senzilla utilitzant amb comandes **curl**[15]. Primer, necessitem obtenir un token per assegurar-nos que tenim permís per enviar sol·licituds al servidor de Logmeal. Després, hem d'utilitzar l'opció **-F**, que indica que farem servir el format **multipart/form-data**. A la part del body de la sol·licitud, hem d'omplir amb el contingut generat pel SDK i llavors ja podrem enviar la sol·licitud.

```
curl -X POST \
  -H "Authorization: Bearer YOUR_AUTH_TOKEN" \
  -H "Accept: application/json" \
  -F "image=@/path/to/your/image.jpg" \
  -F "sequence=@/path/to/your/sequence" \
  -F "preconsumption_image_id=YOUR_PRECONSUMPTION_ID" \
  -F "cam_focal_length=YOUR_CAM_FOCAL_LENGTH" \
  -F "cam_principal_point=YOUR_CAM_PRINCIPAL_POINT" \
  -F "main_rgb_idx=YOUR_MAIN_RGB_IDX" \
  -F "camera_pose=YOUR_CAMERA_POSE" \
  -F "version=YOUR_VERSION" \
  YOUR_ENDPOINT_URL
```

2.6 Resultats

En aquesta secció, es presentarem i analitzaran críticament tots els resultats obtinguts, tenint en compte si el projecte ha assolit els requisits i objectius establerts a la secció 2.1.

En primer lloc, cal assenyalar que l'objectiu principal del projecte és crear un SDK per a un model d'estimació de volum i quantitat d'aliments. Aquest SDK ha de recopilar informació de fotogrames de càmera, classificar i processar aquests fotogrames mitjançant l'anàlisi del centre de l'escena i dels angles formats entre el primer fotograma i el centre. Així es capturen les imatges necessàries per construir la representació dels models d'aliments. També s'integren i es proporcionen les eines necessàries per cridar els serveis de l'API LogMeal. Aquest objectiu s'ha assolit amb èxit, ja que s'han pogut generar tots els fitxers necessaris. Amb aquests fitxers, ja es pot crear un model d'estimació de volum i quantitat d'aliments, complint així els requisits establerts al començament del projecte.

A continuació, des 2 punts de vista, expliquem els resultats obtinguts:

1. **Resultats de l'algoritme SDK**: Observant la figura següent Fig. 2.19, es pot veure clarament com el SDK recull informació de cada fotograma de la càmera, calculant en temps real el centre de l'escena, l'angle entre el fotograma actual i el primer, i recopilant 50 fotogrames dins del rang especificat. D'aquests 50 fotogrames, es pot apreciar clarament que la trajectòria del moviment de les imatges segueix un patró d'angle concret, la qual cosa demostra l'èxit del nostre algoritme. A més del propi moviment, s'han generat amb èxit arxius JSON, seqüència de vídeo i la imatge principal, satisfent els requisits de l'API. En la següent secció de resultat, mostrarem els resultats concrets obtinguts de l'API.

[15]Curl és una eina de comandes molt utilitzada per transferir dades mitjançant diversos protocols

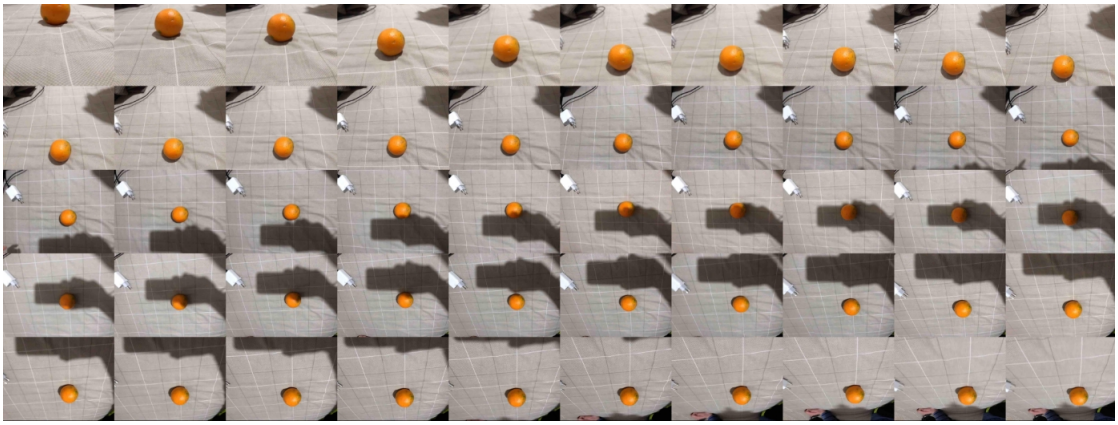

Figure 2.19: Fotogrames clau que formen per vídeo.

Hem aconseguit implementar amb èxit de l'SDK i ja l'hem pujat a **CocoaPods**[16]. Ara, qualsevol desenvolupador pot descarregar i fer servir el nostre SDK a través de CocoaPods. Només cal utilitzar **pod 'LogMealDepth'**[17] per descarregar el nostre SDK[18] i integrar-lo al vostre projecte.

2. **Resultats obtinguts de l'API**: Amb només completar els algoritmes del SDK no és suficient, necessitem integrar el nostre SDK amb l'API de LogMeal. A continuació, el nostre SDK obtindran el següent image Fig. 2.20 i generaran els fitxers necessaris per al servidor, després enviant una sol·licitud al servidor de l'API de LogMeal.

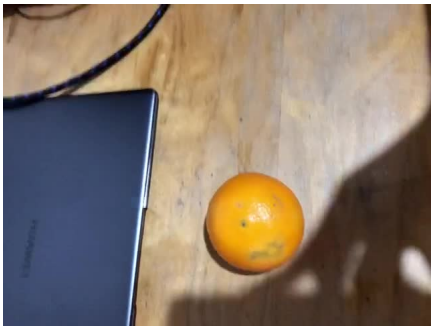

Figure 2.20: Una imatge de taronja que s'envia a l'API.

Després d'un temps, un cop el servidor ha processat les sol·licituds, rebrem una resposta. Amb aquesta resposta, podrem veure clarament que les imatges preses amb l'SDK i els fitxers generats poden ser reconeguts amb èxit per l'API de LogMeal com a taronja.


```
"foodFamily":[
   {
      "id": 6,
      "name": "fruit",
      "prob": 0.82325030
```


[16]CocoaPods és un gestor de dependències per a projectes de desenvolupament d'aplicacions per a iOS i macOS. Facilita la integració de llibreries i frameworks externs en els teus projectes de manera senzilla i eficient.

[17]pod 'LogMealDepth' és una comanda que s'utilitza en un fitxer Podfile per indicar a CocoaPods que descarregui i integri una llibreria anomenada "LogMealDepth" al teu projecte.

[18]`https://gitlab.com/logmeal/logmeal-depth-sdk-ios`

```
    }
    "imageId": 1530123,
    "processed_image_size": {
        "height": 360,
        "width": 480
    },
    "segmentation_results": [
        {
            "center": {
                "x": 321,
                "y": 230
            },
            "contained_bbox": {
                "h": 162,
                "w": 174,
                "x": 234,
                "y": 152
            },
            "recognition_results": [
                "foodFamily": [
                    {
                        "id": 6,
                        "name": "fruit"
                    }
                ],
                "foodType": {
                    "id": 2,
                    "name": "ingredients"
                },
                "hasNutriScore": true,
                "id": 12563,
                "name": "orange",
                "nutri_score": {
                    "nutri_score_category": "A",
                    "nutri_score_standardized": 78
                }
            ]
        }
        ...
    ]
]
```


A través d'aquests dos resultats, revisitem la secció 2.1.1 i els objectius inicials. Hem satisfet els requisits funcionals inicialment proposats amb aquest SDK. Ha demostrat la capacitat de recollir informació dels fotogrames de la càmera, classificar-los, processar-los, calcular el centre de l'escena i els angles formats entre el primer fotograma i el centre. Els usuaris poden utilitzar aquest SDK per capturar estimacions del volum i la quantitat dels aliments i han rebut respostes correctament. En aquest procés, hem analitzat detalladament els requisits del sistema, hem realitzat un disseny exhaustiu i hem implementat diversos algoritmes. A més aquest SDK és utilizable per a altres desenvolupadors.

Capítol 3

Part2: Millora de la Xarxa *Image Transformer Network* per al reconeixement i classificació

En segon part de treball, explorarem com optimitzar el reconeixement de menjar via una xarxa neuronal de tipus *Image Transformer Network*[9] per reconeixement i classificació d'aliments amb precisió. Aquest procés inclou l'anàlisi de la investigació prèvia, la construcció d'una xarxa basat en aquesta investigació, l'optimització de la xarxa i la presentació i discussió dels resultats de la xarxa.

Primer, revisarem les investigacions prèvies per entendre les tecnologies i mètodes existents. En aquesta secció, analitzarem detalladament la tesi de màster d'Andrey, detallarem la seva recerca i treballs, ajudant-nos a determinar la millor ruta tecnològica i aclarir els nostres objectius.

A continuació, construirem una Image Transformer Network basat en aquestes investigacions prèvies. Explicarem detalladament el procés de selecció i construcció de la xarxa, incloent-hi la recopilació i preprocés de dades, el disseny de l'arquitectura de la xarxa i la selecció dels mètodes d'entrenament.

Després, realitzarem l'optimització de la xarxa. Aquesta secció explorarà diverses estratègies d'optimització i modificacións d'hiperparàmetres per millorar la precisió i l'eficiència de la xarxa. Discutirem els mètodes d'optimització específics i el seu impacte en el rendiment del model.

Finalment, presentarem els resultats de l'avaluació del model entrenat. Provarem i avaluarem el model segons el conjunt de dades de classificació d'aliments propietat de LogMeal. A més, discutirem el desplegament del model en escenaris reals i la seva aplicació pràctica, analitzant el seu rendiment en entorns reals.

Amb aquests passos, explorarem i implementarem una xarxa eficient de reconeixement d'imatges d'aliments, proporcionant un suport sòlid per a la seva aplicació en la vida real.

3.1 Anàlisi investigacions prèvies

El treball final de màster [10] d'Andrey té com a objectiu abordar els reptes de la classificació d'aliments de gra fi mitjançant la proposta d'una nova arquitectura anomenada **Multi-task Discriminative Expert Heads Assortment (MDEH)**. La investigació se centra en utilitzar el conjunt de dades LogMeal per millorar la precisió de la classificació en aplicacions de visió per computador relacionades amb els aliments. Ara explorem pas a pas l'essència de l'article d'Andrey per veure quins aspectes poden ajudar-nos a reconstruir i optimitzar la xarxa neuronal Image Transformer Network.

3.1.1 Metodologia

Backbone Architecture

Andrey va utilitzar l'arquitectura **CSwin Transformer**[11] Fig. 3.1 com a Backbone Architecture[1]. Aquesta arquitectura combina *mecanismes d'atenció jeràrquica* [15] i *Locally-Enhanced Positional Encoding*[16] per fer front a la complexitat de les imatges d'alta resolució. CSwin Transformer és una arquitectura Transformer avançada per a tasques de visió per computador. Mitjançant la introducció de nous mecanismes d'atenció i mètodes de codificació de posicions, té com a objectiu millorar la capacitat de processament i l'extracció de característiques de les imatges d'alta resolució. A continuació es detalla l'arquitectura CSwin Transformer:

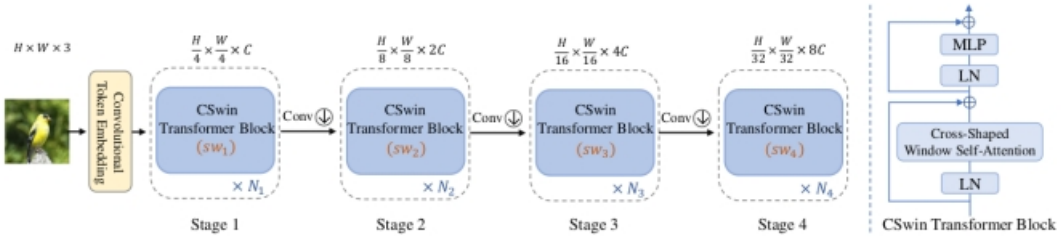

Figure 3.1: Arquitectura del model CSwin. S'exposa les diferents parts que componen aquest model. Font: Dong. (2021).

Les xarxes neuronals convolucionals (CNN) [12] tradicionals extreuen característiques utilitzant filtres convolucionals en regions locals quan processen dades d'imatges. D'altra banda, Transformers[17] van tenir un èxit enorme inicialment en el camp del processament del llenguatge natural (NLP) [13], amb el seu mecanisme d'autoatenció (Self-Attention Mechanism) [14] que permet modelar les relacions de dependència entre diferents posicions a escala global. CSwin Transformer combina l'extracció de característiques locals de CNN amb la capacitat de modelització global de Transformers, oferint una nova manera de tractar les imatges d'alta resolució.

[1]Estructura principal que serveix com a base per al disseny i desenvolupament de sistemes.

Mecanismes d'atenció jeràrquica

La innovació central de CSwin Transformer rau en el seu mecanisme d'atenció jeràrquic Fig. 3.2, que, a través de l'ampliació progressiva de l'àmbit d'atenció, permet l'extracció de característiques des de local a global.

- **Atenció local autònoma**: Primer, la imatge d'entrada es divideix en petits blocs (patches), i després s'aplica un mecanisme d'autoatenció a cada petit bloc. Aquesta tècnica ajuda a extreure característiques detallades en un àmbit local.
- **Atenció horitzontal i vertical**: Es divideix la representació de característiques d'entrada en bandes horitzontals i verticals, i s'aplica l'atenció a cadascuna d'aquestes bandes. Aquest mètode millora la captura de característiques en diferents direccions, al mateix temps que redueix la complexitat computacional.
- **Atenció global autònoma**: A nivells superiors, la dimensionalitat de la representació de característiques es redueix progressivament, ampliant l'abast de l'atenció mitjançant la superposició de finestres d'atenció per integrar progressivament les característiques de local a global. Gràcies a aquest mecanisme d'atenció jeràrquic, CSwin Transformer pot processar eficaçment imatges d'alta resolució i extreure característiques útils tant en àmbits locals com globals.

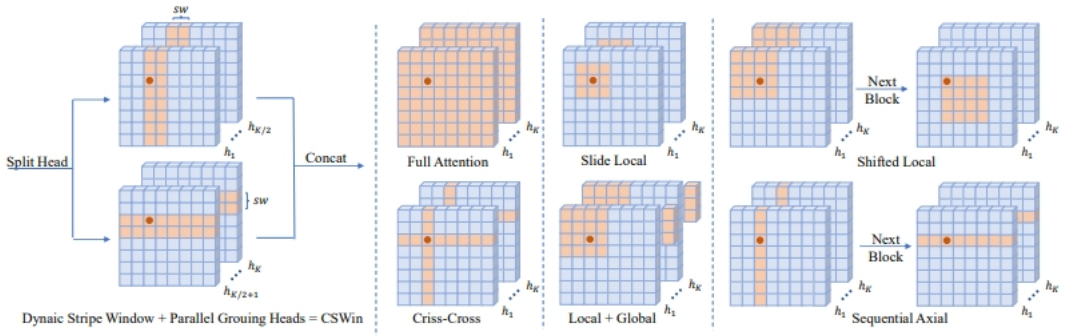

Figure 3.2: Il·lustració de diferents mecanismes d'autoatenció. El CSWin és fonamentalment diferent en dos aspectes. Primer, dividim els multi-heads ($h1 . . . . . hk$) en dos grups i realitzem l'autoatenció en ratlles horitzontals i verticals simultàniament. En segon lloc, ajustem l'amplada de les ratlles segons la profunditat de la xarxa, aconseguint un millor compromís entre el cost de càlcul i la capacitat. Font: Dong (2021).

Locally-Enhanced Positional Encoding (LePE):

El CSwin Transformer introdueix un nou mètode de codificació de posicions anomenat **Locally-Enhanced Positional Encoding (LePE)** Fig. 3.9. Els mètodes tradicionals de codificació de posicions absolutes poden tenir limitacions en el tractament d'imatges d'alta resolució, mentre que el LePE millora l'efecte de la representació de característiques mitjançant l'enfortiment de la informació de posició en àrees locals.

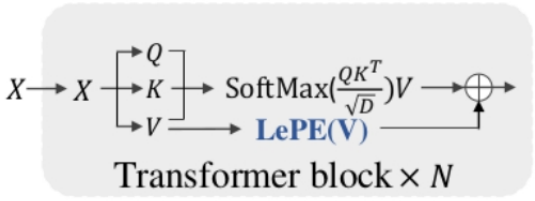

Figure 3.3: Rresentació de LePE positional encoding method. Font: Font Dong. (2021)

Concretament, el LePE s'aplica després del càlcul de l'atenció per millorar la informació de posició a la representació de característiques. Aquest mètode no només conserva els avantatges de la codificació de posicions, sinó que també permet capturar de manera més precisa les característiques de posició en un àmbit local, millorant així el rendiment del model.

Avantatges del CSwin Transformer:

1. **Extracció eficient de característiques**: Gràcies al mecanisme d'atenció jeràrquica, el CSwin Transformer pot extreure característiques de manera eficient tant a nivell local com global, amb una baixa complexitat computacional en el tractament d'imatges d'alta resolució.
2. **Mecanisme d'atenció flexible**: El mecanisme d'atenció jeràrquica proporciona flexibilitat per capturar característiques a diferents escales, permetent adaptar-se millor a diverses tasques de processament d'imatges.
3. **Codificació de posicions millorada**: El mètode de Locally-Enhanced Positional Encoding incrementa l'eficàcia de la representació de característiques, augmentant la precisió del model en tasques sensibles a la posició.

En resum, el CSwin Transformer, mitjançant la introducció del mecanisme d'atenció jeràrquica i la codificació de posicions millorada a nivell local, ofereix un mètode eficient i flexible per al processament d'imatges. Combina els avantatges de les xarxes neuronals convolucionals i Transformer, demostrant un excel·lent rendiment en el tractament d'imatges d'alta resolució i l'extracció de característiques, essent una innovació important en el camp de la visió per computador.

Discriminative Expert Heads

Andrey afegeix els **Discriminative Expert Heads (DEH)** Fig. 3.4 basats en CSwin. Els DEH estan dissenyats per abordar de manera més eficaç la complexitat de les tasques de classificació de granularitat fina, especialment en relació amb diferents tipus d'aliments, begudes i ingredients. Cada DEH està especialment entrenat per gestionar una categoria semàntica específica, com ara aliments, begudes, ingredients i no aliments. Aquesta estructura de múltiples caps permet que el model especialitzi en diferents àrees, millorant així la precisió de la classificació i el rendiment global del model.

Els DEH no només són capaços de classificar objectes dins del seu domini específic, sinó que també poden gestionar adequadament els objectes fora del seu domini. Aquesta capacitat fa que

el model tingui una millor capacitat de generalització, adaptant-se millor a les diferents entrades de dades i mostrant una alta robustesa en la identificació d'objectes desconeguts.

A més, la independència dels DEH facilita el procés d'entrenament i ajust del model. Amb la capacitat de entrenar i ajustar cada cap de manera independent, és possible optimitzar millor el rendiment del model, alhora que s'incrementa la seva flexibilitat i escalabilitat.

En resum, els DEH, com a part important del model, proporcionen una solució efectiva per a la gestió de tasques de classificació de granularitat fina, permetent així un notable augment del rendiment en àrees com la identificació d'aliments.

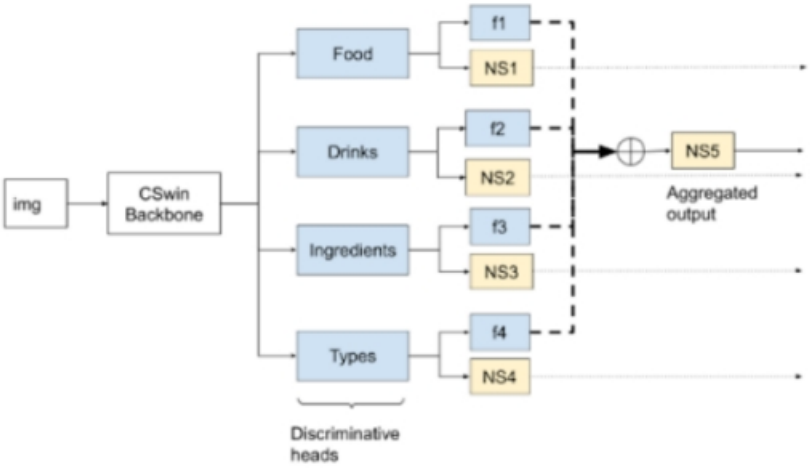

Figure 3.4: Representa les diferents peces que s'ajusten al model. En blau, es representa cada cap expert discriminatiu especialitzat en el domini destacat. Cada cap inclou una sortida de representació de característiques identificada com a 'f_x' i una sortida de classificació identificada com a 'NSx' especificat pel símbol $\oplus$ amb les seves característiques de sortida corresponents. Finalment, la sortida de classificació agregada s'identifica com a '$NS5$'.

Arquitectura de DEH

Cada DEH està compost per les següents parts:

1. **Capa d'entrada**: Rep les representacions de característiques enviades des de la xarxa neural CSwin.
2. **Capa d'extracció de característiques**: Aquesta capa processa encara més les característiques d'entrada i extreu les representacions de característiques específiques del domini en qüestió. El seu disseny està basat en la quarta fase del model CSwin, però s'ajusta per adaptar-se a les necessitats de diferents categories semàntiques.
3. **Capa de classificació**: Utilitza una capa de **Normed-Softmax** per a la classificació. Aquesta capa no només pot gestionar el problema de desequilibri de categories, sinó que també pot millorar la precisió de classificació de les categories minoritàries. La capa de classificació produeix una distribució de probabilitat de classificació multi-classe, que inclou totes les categories dins d'aquell domini i una categoria "altres".

Funcionament dels DEHs és el següent

1. **Extracció de característiques**: La imatge d'entrada passa a través de les primeres capes del model CSwin per a l'extracció de característiques, generant una representació de característiques d'alta dimensió. Aquestes representacions es transfereixen a diversos DEHs.
2. **Classificació de domini**: Cada DEH rep aquestes característiques i les processa per a la classificació específica del seu domini. Per exemple, un "DEH d'aliments" pot classificar la imatge d'entrada en categories específiques d'aliments.
3. **Identificació fora de domini**: Dins de cada DEH, el model intenta identificar si la imatge d'entrada no pertany al seu domini. Si és així, es classificarà com a "altres".
4. **Agregació de característiques**: Les representacions de característiques extretes de cada DEH es combinen per a crear una representació de característiques global. Aquesta representació global s'utilitza per a la tasca final de classificació, millorant així la precisió i la robustesa de la classificació global.

En conclusió, els DEHs són una part més important del model proposat en l'article. Amb la seva concentració en tasques de classificació de diferents categories semàntiques, han millorat notablement l'eficàcia de la classificació de menjar de granularitat fina. La combinació de classificació dins del domini i identificació fora del domini permet als DEHs gestionar millor tasques de classificació complexes i millorar el rendiment global del model.

3.1.2 Extracció i Agregació de Característiques Multi-task

Aquesta arquitectura inclou un pas d'agregació que combina les representacions de característiques de diversos DEH per generar una representació global, utilitzada per a la tasca final de classificació. En el treball d'Andrey, l'extracció i l'agregació de característiques són part central de l'arquitectura dels **Multi-task Discriminative Expert Heads (MDEH)** Fig. 3.6 proposats.

- **Extracció de Característiques**
  - Arquitectura Base:
    1. Utilitza l'arquitectura CSwin Transformer com a base per a l'extracció de característiques. Aquesta arquitectura processa imatges d'alta resolució mitjançant un mecanisme d'atenció jeràrquic i LePE.
  - Discriminative Expert Heads
    1. El model utilitza diversos DEH específics de domini, cadascun centrat en diferents categories semàntiques (com ara menjar, beguda, ingredients i no-menjar).
    2. Cada DEH és responsable de classificar detalladament els objectes dins del seu domini i de detectar els objectes fora d'aquest. Les sortides de classificació de cada DEH inclouen etiquetes de granularitat fina dins del domini, així com una etiqueta "altres" que representa els objectes fora del domini.

- **Pas d'Agregació**

  1. Les representacions de característiques generades per cada DEH es transmeten al pas d'agregació. Aquest pas combina les representacions de característiques de tots els dominis per formar una representació de característiques global.
  2. El procés d'agregació de característiques es realitza mitjançant la mitjana de les representacions de característiques de cada DEH. La fórmula és la següent:

$$f = \frac{1}{K+1} \sum_{0 \leq k \leq K+1} fk$$

  On fk és és la representació de característiques de cada cap, K és el nombre de DEH, i finalment s'obté la representació de característiques global f.

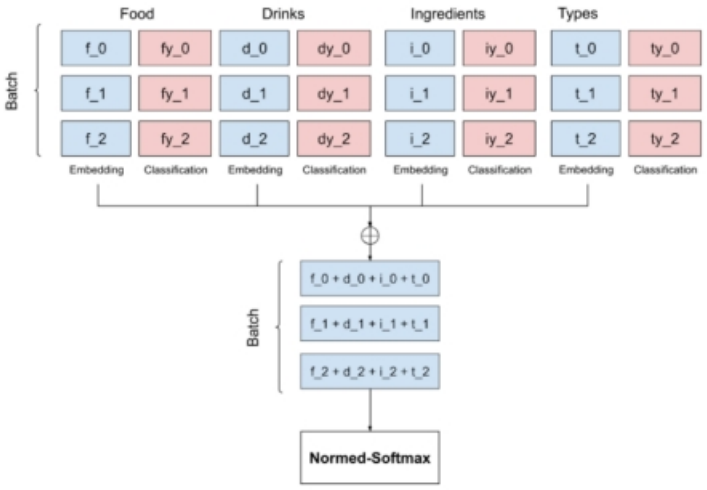

Figure 3.5: Repassa el pas d'agregació de funcions, destacant la incorporació de les incrustacions de tasques com a representació generalitzada.

- Classificació Final:

  1. La representació de característiques global agregada es transmet a la capa de classificació final. La tasca d'aquesta capa és realitzar la classificació multiclasse final dels objectes, cobrint totes les etiquetes de granularitat fina del conjunt de dades, així com una etiqueta "*non food*" per identificar objectes fora del domini.
  2. Per abordar el desequilibri de classes, la capa de classificació utilitza la **Normed-Softmax** en lloc de la Softmax tradicional. La Normed-Softmax normalitza els pesos per augmentar la sensibilitat cap a les classes minoritàries.

$$\text{Normed Softmax}(x_i) = \frac{e^{s(x_i - \mu)}}{\sum_j e^{s(x_j - \mu)}}$$

  On x_i és el i-èssim element del vector d'entrada, és μ la mitjana del vector d'entrada, s és un paràmetre ajustable que controla la força de la normalització.

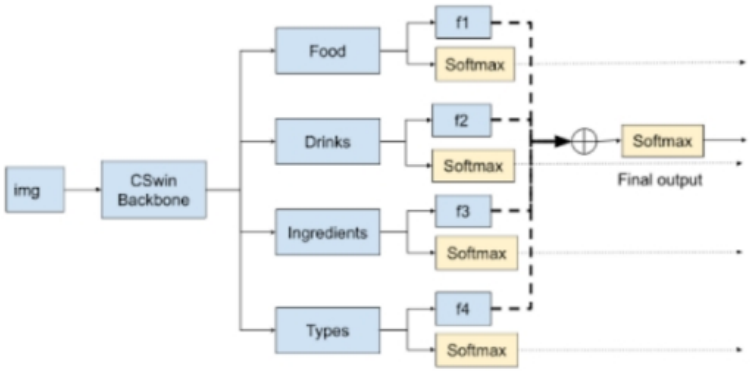

Figure 3.6: Articula la disposició del model base MDEH.

Tècniques de Regularització

Per abordar els problemes de desequilibri de classes i sobreajustament, s'ha implementat la Normed-Softmax a la capa de classificació. Aquesta capa normalitza els pesos per incrementar la sensibilitat cap a les classes minoritàries.

En cada capa de classificació s'utilitza la Normed-Softmax per abordar el desequilibri de classes. Aquesta tècnica normalitza els pesos per millorar la capacitat del model per identificar les classes minoritàries, evitant així la situació en què les classes minoritàries són ignorades com freqüentment ocorre amb la Softmax tradicional.

El diagrama Fig. 3.7 mostra l'estructura del model utilitzant la Normed-Softmax, amb la capa de classificació actualitzada a la dreta després de la implementació de la Normed-Softmax.

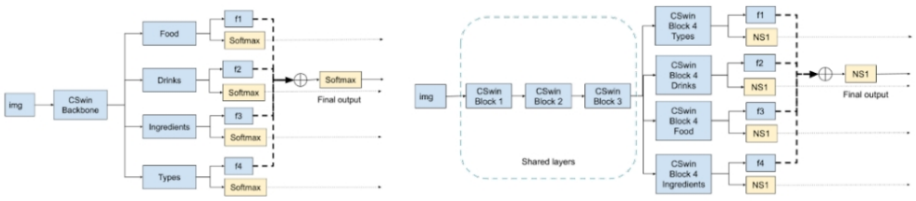

Figure 3.7: l'estructura del model utilitzant la Normed-Softmax.

Funció de Pèrdua

S'ha utilitzat una funció de pèrdua multi-tasca que combina la pèrdua de cada DEH amb la pèrdua de classificació agregada. Això assegura un entrenament equilibrat per a cada tasca.

La funció de pèrdua multi-tasca combina les pèrdues de cada DEH amb la pèrdua de classificació final. Això garanteix un entrenament equilibrat per a totes les tasques, permetent al model centrar-se simultàniament en les tasques de classificació de cada domini. La fórmula de la funció de pèrdua és la següent:

$$loss = loss_{drinks} + loss_{food} + loss_{ingredients} + loss_{types} + loss_{aggr}$$

Aquest disseny de la funció de pèrdua assegura que el model no mostrisca preferència per cap tasca específica durant el procés d'entrenament, sinó que tingui en compte el rendiment de totes les tasques de manera global.

L'extracció i agregació de característiques són components clau de l'arquitectura MDEH proposada en treball d'Andrey. Mitjançant l'aprenentatge multitasca i els DEHs específics de cada domini, s'extreuen característiques de diferents àmbits i s'agreguen en un únic vector de característiques per aconseguir una classificació precisa de diverses classes. L'ús de la capa Normed-Softmax i la funció de pèrdua multi-tasca aborden eficaçment els problemes de desequilibri de classes i sobreajustament, millorant així el rendiment global del model.

3.1.3 Conjunt de dades

El projecte utilitza el conjunt de dades personalitzat de LogMeal Fig. 3.8, que conté més de 1.348.507 imatges. Aquestes imatges estan distribuïdes en més de 800 categories d'aliments diferents, que inclouen cuines com la d'Itàlia, França, els Països Baixos i moltes altres. El conjunt de dades està dividit en quatre categories semàntiques: non-food, Ingredients, Drinks i Food, i cadascuna d'aquestes categories es divideix en etiquetes més específiques.

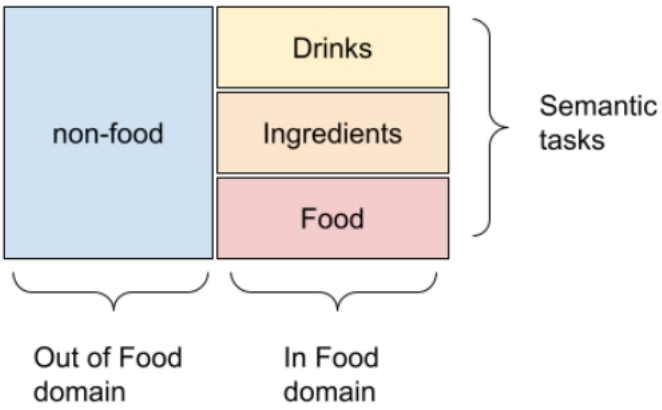

Figure 3.8: Conjunts de dades de fora de domini(non-food) i conjunts de dins de dominis considerats en les seves respectives precisions non-food i dins del domini.

3.1.4 Resultat

La secció experimental destaca la precisió millorada de l'arquitectura MDEH:

Precisió: Va aconseguir la màxima precisió top-1 del **81.47** i un f1-score de **76.52**. Aquest model va utilitzar la capa Normed-Softmax per a la regularització, mostrant millores significatives respecte als models anteriors (Hydra amb un increment del 6.62% i CSwin amb un increment del 33.75.

3.1.5 Conclusió:

A través dels resultats d'Andrey, ho podem saber:

1. **Tècniques de regularització**: La capa Normed-Softmax ha obtingut resultats destacats en la regularització, augmentant la sensibilitat del model envers les classes minoritàries i millorant significativament la seva precisió, especialment en el tractament dels problemes de desequilibri de dades.
2. **Tractament del desequilibri de dades**: La precisió del model depèn en gran mesura de l'ús efectiu de mètodes per abordar el desequilibri de dades, especialment en escenaris d'aprenentatge multi-tasca.
3. **Arquitectura multi-tasca**: Aquesta arquitectura permet una millor extracció de característiques i generalització, cosa que fa que el model rendeixi bé en diferents dominis i millora la capacitat de manejar elements no alimentaris simultàniament..

És evident que aquests tres factors han contribuït significativament a la millora dels resultats, cosa que ens entusiasma molt. En el nostre següent projecte, tenim la intenció de centrar-nos en aquests tres aspectes clau per reproduir el model d'Andrey i analitzar quin factor té l'impacte més gran en els resultats. A més, investigarem possibles punts d'optimització per tal de perfeccionar encara més el model.

3.2 Objectiu de la II Part del Treball

Els resultats de les investigacions d'Andrey han revelat tres factors clau que han impulsat significativament el rendiment dels transformadors d'imatges: tècniques de regularització, tractament del desequilibri de dades i arquitectura multi-tasca. Aquestes troballes ens han motivat a emprendre un projecte per reproduir el seu model i analitzar l'impacte relatiu d'aquests factors.

L'objectiu d'aquesta part del projecte és profunditzar com els Transformers aplicats sobre imatges per produir una millora de les implementacions actuals. Concretament, ens centrarem en els següents passos:

1. **Reconstrucció del Model**: Implementarem el model proposat per Andrey amb precisió, incloent-hi les tècniques de regularització, el tractament del desequilibri de dades i l'arquitectura multi-tasca.
2. **Anàlisi d'Impacte**: Realitzarem experiments per avaluar l'efecte individual de cadascun dels factors identificats per Andrey. Compararem la precisió del model amb i sense cadascuna d'aquestes tècniques per determinar-ne la relativa importància.
3. **Optimització Contínua**: Basant-nos en els resultats de l'anàlisi d'impacte, identificarem àrees específiques per a la millora contínua. Això pot incloure ajustaments als hiperparàmetres, l'exploració de noves tècniques de regularització o refinaments en l'arquitectura del model.

A través d'aquest procés, buscarem millorar tant la precisió com l'eficiència de la xarxa Transformer d'imatges, obrint noves oportunitats per a l'aplicació d'aquesta tecnologia en una àmplia gamma de tasques de processament d'imatges.

3.3 Reconstruir el model

Donat el problema del reconeixement de mejar, els focatlitezm en la recollida i el preprocessament de dades, la selecció del model, els mètodes d'entrenament i la modificació dels hiperparàmetres.

3.3.1 Recollida i preprocesament de dades

Hem d'utilitzar el conjunt de dades de LogMeal, presentem una classe anomenada **LogMealMergedDatasetV3**, que hereta de la classe Dataset. L'objectiu principal d'aquesta classe és gestionar i processar el conjunt de dades per a ser utilitzat en l'entrenament i la prova de models d'aprenentatge automàtic.

1. Generar les base dades necessàries i els fitxers corresponents.
2. Processar cada fitxer de les etiquetes i agregar-los mitjançant una unió externa segons la columna id.
3. Processar les dades no relacionades amb els aliments i connectar-les amb el conjunt de dades.
4. Omplir les etiquetes que falten amb l'etiqueta *Altres* i convertir totes les etiquetes a nombres enters.
5. Assignar etiquetes segons el tipus de conjunt de dades.
6. Calcular la distribució i el nombre de classes per a cada fitxer de dades.
7. Calcular la distribució i el nombre de classes de les etiquetes agregades i generar la funció de transformació corresponent.
8. Carregar les imatges i preprocesar la mida de les imatges, retornant un diccionari amb les dades de les imatges i les etiquetes.

En resum, la funcionalitat principal d'aquesta classe és llegir, agregar, processar les dades i proporcionar funcionalitats convenient per carregar imatges i les seves etiquetes corresponents per a l'entrenament i la prova de models d'aprenentatge automàtic.

En segon lloc, presentem una altra classe anomenada **LogMealTypesDataset**. Aquesta classe té com a objectiu gestionar i processar dades amb imatges i etiquetes, especialment per a la preparació de conjunts de dades per a tasques de classificació d'imatges, i proporcionar algunes funcions d'augmentació i transformació:

1. Actualitzar les etiquetes de les mostres i les categories agregacions.
2. Eliminar les categories i les mostres etiquetades per a l'eliminació.

Mitjançant la filtració i modificació del conjunt de dades, es pot millorar significativament la qualitat de les dades, equilibrar la distribució de les dades, simplificar el conjunt de dades, millorar la capacitat de generalització del model i generar conjunts de dades que compleixin els requisits específics de l'experiment. Aquests passos són fonamentals per construir models d'aprenentatge automàtic eficients, precisos i robustos.

La funcionalitat principal de la classe **LogMealTypesDataset** és gestionar i processar conjunts de dades per a la classificació d'imatges. Llegeix les rutes de les imatges i les etiquetes, suporta operacions d'augmentació i transformació de dades, pot gestionar diferents modificacions (com ara eliminar mostres, reetiquetar mostres, agregar i eliminar categories). Això fa que sigui adequada per a tasques de classificació d'imatges complexes i proporciona dades de alta qualitat per a l'entrenament i l'avaluació de models d'aprenentatge automàtic.

3.3.2 Selecció i Construcció del Model per Reconeixement de Menjar

1. Implementació de capes i blocs bàsics:
   - **Capa MLP**[18]: Un perceptró multi-heads amb activació GELU[19] i Dropout[20].
   - **LePEAttention**: Implementa el mecanisme d'atenció de finestres creuades, utilitzat per al càlcul l'atenció en àrees locals de la imatge.
   - **CSWinBlock**: Combina el mecanisme d'atenció local i global amb la capa MLP per formar un bloc Transformer bàsic i per extracció de característiques de la imatge: extreu progressivament les característiques de la imatge.
   - **Merge Block**: Utilitzat per agregar característiques de diferents resolucions, d'altes a baixes.
   - **Hydra Heads**: Estructura de sortida multi-heads, amb cada head corresponent a una tasca de classificació diferent. gestiona simultàniament multi-task de classificació relacionades. Cada cap pot gestionar independentment diferents tasques de classificació, com ara tipus d'aliments, begudes, aliments, ingredients, etc.
2. Classe CSWinTransformer[2]:
   - Configura els paràmetres estructurals del model, com ara mida de la imatge, mida del fragment, dimensió d'integració, profunditat, mida de la divisió, ràtio MLP, etc.
   - Definició de l'estructura jeràrquica del model inclou la capa d'integració convolucional, diverses etapes (stages) formades per **CSWinBlocks** i el **Merge_Block** per a la agregació de característiques.
   - Classificació multi-heads (Hydra Heads) per a la classificació multi-task, es defineixen múltiples heads de sortida, cadascun responsable d'una tasca de classificació diferent.

[2]https://github.com/microsoft/CSWin-Transformer

3.3.3 Mètodes d'entrenament

Hem definit una funció *train_epoch* per entrenar un model, que executa tots els passos necessaris durant un cicle d'entrenament.

Passos principals

1. En el pre-processament de les dades, incloent-hi el tractament **Mixup**. Hem implementat la tècnica Mixup per realitzar una augmentació de dades durant la fase de pre-processament. Mixup és una tècnica que combina imatges i les seves etiquetes en una proporció determinada per generar noves mostres d'entrenament. Aquesta tècnica no només millora la capacitat de generalització del model i redueix el sobreajustament, sinó que també promou la robustesa del model davant de variacions del conjunt de dades. A continuació, detallem la nostra configuració de Mixup:

Algorithm 8 Perform Mixup

```
function PerformMixup(b_, num_classes, ds, mixup_fn)
    idxs ← (b_[ds] < num_classes − 1)
    if idxs.sum() mod 2 == 1 then
        idxs[idxs.nonzero(as_tuple=True)[0][-1]] ← False
    end if
    placeholder ← torch.zeros_like(torch.empty(b_[ds].__len__(), num_classes))
    mix_imgs, mix_lbls ← mixup_fn['mixup'](b_[img][idxs], b_[ds][idxs])
    placeholder[idxs, : −1] ← mix_lbls
    holder_idxs ← placeholder.nonzero(as_tuple=True)
    aggr_idxs ← placeholder[idxs].nonzero()[:, 1] + mixup_fn['conv_dists'][0]
    b_[aggr][(holder_idxs[0], aggr_idxs)] ← placeholder[holder_idxs]
    placeholder[∼ idxs, −1] ← 1
    b_[ds] ← placeholder
    b_[img][idxs] ← mix_imgs
end function
```

Primer, seleccionem les mostres que necessiten el tractament Mixup segons certes condicions, assegurant-nos que les mostres estiguin aparellades. Després, executem l'operació Mixup per generar imatges i etiquetes combinades. Finalment, assignem els resultats combinats a les dades originals.

Algorithm 9 Mixup Configuration

```
function MixupConfiguration(datasets, num_classes, conv_dists)
    conf ← {}
    mixup_args ← {
        'mixup_alpha' : 50.,
        'cutmix_alpha' : 0.,
        'cutmix_minmax' : None,
        'prob' : 1.0,
        'switch_prob' : 0.,
        'mode' :' batch',
        'label_smoothing' : 0,
    }
    for entry, ds in enumerate(datasets) do
        mixup_args['num_classes'] ← num_classes[ds] − 1
        mixup_fn ← Mixup(**mixup_args)
        conf[ds] ← {'mixup' : mixup_fn}
        conf[ds]['conv_dists'] ← [conv_dists[entry], conv_dists[entry+1]]
    end for
    return conf
end function
```

2. Transferir les dades a la GPU: Aquest pas transfereix les dades preprocessades des de la CPU a la GPU per aprofitar la potència de càlcul de la GPU i accelerar el procés d'entrenament[21].

3. Propagació endavant per calcular les sortides del model: En aquesta fase, introduïm les dades al model per calcular les sortides. Després, calculem les pèrdues per a diferents tasques (com aliments, begudes, ingredients, etc.). Hem utilitzat la funció **Normad-softmax** i **focal_loss**, que és una versió millorada de la pèrdua d'entropia creuada dissenyada per abordar el problema del desequilibri de classes. Aquestes funcions introdueixen dos hiperparàmetres, i , per ajustar l'enfocament en les mostres difícils de classificar i les fàcils de classificar.

   Aquests són els mètodes d'implementació de **Focal loss** i **Normad-softmax**

Algorithm 10 Focal Loss Calculation

```
function FocalLoss(labels, logits, α, γ, is_softlabel=False)
    BCLoss ← F.binary_cross_entropy_with_logits(input=logits, target=labels, reduce=False)
    if γ == 0.0 then
        modulator ← 1.0
    else
        modulator ← exp(−γ · labels · logits − γ · log(1 + exp(−1.0 · logits)))
    end if
    loss ← modulator · BCLoss
    weighted_loss ← α · loss
    focal_loss ← ∑(weighted_loss) / ∑(labels)
    return focal_loss
end function
```

Algorithm 11 Normed-Softmax Balanced Loss Calculation

function CB_Loss(labels, logits, samples_per_cls, no_of_classes, β, γ)
    effective_num $\leftarrow 1.0 - (\beta^{\text{samples\_per\_cls}})$
    weights $\leftarrow (1.0 - \beta)/\text{array}(\text{effective\_num})$
    weights[weights == inf] $\leftarrow 0$
    weights $\leftarrow$ weights/np.sum(weights) $\times$ no_of_classes
    labels_one_hot $\leftarrow$ F.one_hot(labels, num_classes = no_of_classes).float()
    weights $\leftarrow$ torch.tensor(weights)()
    weights $\leftarrow$ weights.unsqueeze(0).cuda()
    weights $\leftarrow$ weights.repeat(labels_one_hot.shape[0], 1) $\times$ labels_one_hot
    weights $\leftarrow$ weights.sum(1)
    weights $\leftarrow$ weights.unsqueeze(1)
    weights $\leftarrow$ weights.repeat(1, no_of_classes)
    pred $\leftarrow$ logits.softmax(dim = 1)
    cb_loss $\leftarrow$ F.binary_cross_entropy(input=pred, target=labels_one_hot, weight=weights)
    return cb_loss
end function

- Mitjançant la propagació enrere i l'actualització de l'optimitzador, modifican els paràmetres del model per millorar el seu rendiment en les dades d'entrenament.
- Utilitzar entrenament amb precisió mixta per millorar l'eficiència d'entrenament i reduir l'ús de memòria.
- Equilibrar l'aprenentatge de diferents tasques mitjançant l'ajustament de pesos adaptatius i la normalització de gradients, evitant que una tasca domini el procés d'entrenament, millorant així l'eficàcia de l'aprenentatge multi-task.
- Mantenir l'estabilitat numèrica mitjançant l'escalat i el desescalat de gradients, prevenint l'explosió o desaparició de gradients.

4. Calcular **Gradnorm**[23] per a cada tasca i realitzar la normalització de gradients per assegurar que les velocitats d'aprenentatge de les diferents tasques es mantinguin equilibrades.
5. Al final de cada cicle d'entrenament, s'actualitza la taxa d'aprenentatge per ajustar dinàmicament la taxa d'aprenentatge, ajudant el model a convergir millor.
6. Finalment, es calculen i registren contínuament les pèrdues mitjanes, la taxa d'aprenentatge i la precisió d'entrenament del cicle actual per avaluar i ajustar el procés d'entrenament.

Mitjançant aquests passos, la funció *train_epoch* aconsegueix un procés d'entrenament multi-task eficient i estable, permetent que el model obtingui resultats en múltiples tasques simultàniament.

3.3.4 Determinació dels hiperparàmetres

Per entrenar un model nostre, conté múltiples configuracions d'hiperparàmetres. A continuació es presenten els detalls d'aquests hiperparàmetres, incloent-hi la seva funció, utilitat i importància:

1. –lr: Estableix la taxa d'aprenentatge, controlant la mida dels passos en l'actualització dels paràmetres del model.

2. –lr-cycle-mul: Controla l'amplitud del canvi en els cicles de la taxa d'aprenentatge.
3. –opt: Selecciona el tipus d'optimitzador.
4. –cooldown-epochs: Defineix els cicles de refredament, especificant el nombre de cicles d'entrenament en què la taxa d'aprenentatge es manté constant després d'haver finalitzat el programador de taxa d'aprenentatge.
5. –epochs: Nombre total de cicles d'entrenament.
6. –warmup-epochs: Nombre de cicles de preescalfament, on la taxa d'aprenentatge augmenta gradualment.
7. –decay-epochs: Nombre de cicles abans que comenci la disminució de la taxa d'aprenentatge.
8. –decay-rate: Freqüència de disminució de la taxa d'aprenentatge, un factor que es multiplica a la taxa d'aprenentatge en cada disminució.
9. –lr-cycle-limit: Límits del nombre de cicles de la taxa d'aprenentatge.
10. –use_types: Activa la tasca de classificació de tipus.
11. –use_nonfood: Activa la tasca de classificació de no aliments.
12. –loss: Especifica el tipus de funció de pèrdua.
13. –tasks: Nombre de tasques, indicant el nombre de tasques diferents que el model ha de gestionar simultàniament.
14. –pretrained: Utilitza un model preentrenat.
15. –normed-linear: Utilitza capes lineals normalitzades.
16. –use_aggr: Activa la tasca de classificació agregada.
17. –keep-norm: Manté la normalització.
18. –aggregate: Realitza l'operació d'agregació.
19. –no-resume-opt: No restaura l'estat de l'optimitzador.
20. –cutmix: Estableix el paràmetre alpha per a l'augmentació de dades CutMix.
21. –mixup: Estableix el paràmetre alpha per a l'augmentació de dades Mixup.
22. –model: Especifica el nom del model a utilitzar.
23. –data: Ruta del conjunt de dades.
24. –no-prefetch: Desactiva la preextracció de dades.
25. –batch-size: Nombre de mostres entrenades simultàniament en cada pas.
26. –img-size: Mida de la imatge d'entrada.
27. –dataset_type: Tipus de conjunt de dades.
28. –gradient_accumulation: Acumulació de gradients, nombre de gradients acumulats abans d'una actualització de paràmetres.

29. –initial-checkpoint: Ruta del punt de control inicial, utilitzat per carregar els pesos preentrenats.

Hiperparàmetres Importants:

1. Paràmetres relacionats amb la taxa d'aprenentatge (**lr**, **lr-cycle-mul**, **decay-epochs**, **decay-rate**): Aquests paràmetres afecten directament la velocitat i estabilitat de l'entrenament del model. Si la taxa d'aprenentatge és massa alta, l'entrenament pot ser inestable, mentre que si és massa baixa, l'entrenament pot ser molt lent.
2. Optimitzador (opt): Triar un optimitzador adequat pot accelerar la convergència del model i millorar el rendiment. Per example **AdamW** és un optimitzador utilitzem per l'entrenaments.
3. Nombre d'èpoques i mida del lot (epochs, batch-size): El nombre d'èpoques determina la durada total de l'entrenament, mentre que la mida de les mostres entrenades afecta el nombre d'actualitzacions per època i l'ús de la memòria.
4. Augmentació de dades (cutmix, mixup): Aquests paràmetres utilitzen tècniques d'augmentació de dades per millorar la capacitat de generalització del model i reduir el sobreajustament.
5. Paràmetres relacionats amb el model i les dades (model, data, img-size, dataset_type): Aquests paràmetres especifiquen el tipus de model a entrenar, la ruta de les dades, la mida de les imatges i el tipus de conjunt de dades, constituint la base de tot el procés d'entrenament.
6. Preentrenament i punts de control (pretrained, initial-checkpoint): Utilitzar models preentrenats pot accelerar la convergència i millorar el rendiment del model; els punts de control inicials es fan servir per carregar pesos de models existents.

Aquests hiperparàmetres determinen conjuntament diversos aspectes de l'entrenament del model, incloent-hi l'eficiència, el rendiment, l'estabilitat i l'ús dels recursos.

3.4 Optimització del model

Per a l'optimització del model, hem utilitzat una sèrie d'hiperparàmetres específics per entrenar el nostre model *cswin_tiny_224_hydra*. A continuació es descriuen els paràmetres més importants que hem utilitzat, la raó de la seva selecció, així com els objectius i resultats esperats.

Hiperparàmetres Utilitzats:

1. Taxa d'aprenentatge (**–lr 0.0001**): La taxa d'aprenentatge controla la magnitud dels ajustaments als paràmetres del model durant l'entrenament. Hem seleccionat una taxa baixa per assegurar-nos que l'entrenament sigui estable i evitar grans oscil·lacions en la funció de pèrdua.
2. Optimitzador (**–opt adamw**): Hem utilitzat AdamW com a optimitzador per les seves propietats de convergència ràpida i millor regularització dels pesos. AdamW[24] combina

els avantatges de l'optimitzador Adam[25] amb la penalització de pes de L2, millorant el rendiment general del model.

3. Preescalfament i Refredament (**–warmup-epochs 2**, **–cooldown-epochs 0**): La fase de preescalfament durant 2 èpoques ajuda a estabilitzar l'entrenament augmentant gradualment la taxa d'aprenentatge des d'un valor inicial més baix. No s'ha utilitzat una fase de refredament (cooldown-epochs 0) perquè volem mantenir la taxa d'aprenentatge constant després d'arribar al seu valor màxim.

4. Cicles de la taxa d'aprenentatge (**–lr-cycle-mul 1**, **–lr-cycle-limit 4**): Hem configurat la taxa d'aprenentatge per variar cíclicament per evitar quedar atrapats en minims locals. Els cicles múltiples ajuden el model a explorar millor l'espai de paràmetres.

5. Tipus de pèrdua (**–loss mixup**): La pèrdua mixup és utilitzada per millorar la robustesa del model contra el sobreajustament. Això s'aconsegueix combinant parells d'exemples d'entrenament i les seves etiquetes.

6. Augmentació de dades (**–cutmix 1.0**, **–mixup 0.8**): Utilitzem tècniques d'augmentació de dades com CutMix i Mixup per millorar la capacitat de generalització del model. CutMix[26] barreja regions de diferents imatges mentre que Mixup barreja píxels de dues imatges.

7. Tipus de tasques (**–use_types**, **–use_nonfood**): El model s'entrena per a diferents tasques de classificació com tipus d'aliments i elements no alimentaris, la qual cosa requereix el processament de múltiples tasques simultànies.

8. Hiperparàmetres del model (**–model cswin_tiny_224_hydra**, **–normed-linear**, **–use_aggr**, **–keep-norm**, **–aggregate**): El model *cswin_tiny_224_hydra* és especialment adequat per a tasques de visió amb múltiples capçals de sortida. L'ús de capes lineals normalitzades i l'agrupació de resultats millora la precisió i estabilitat del model.

9. Paràmetres d'entrenament (**–batch-size 64**, **–epochs 20**): Hem utilitzat una mida de lot de 64 per assegurar-nos que l'entrenament sigui eficient i les actualitzacions de gradients siguin estables. L'entrenament es realitza durant 20 èpoques per permetre al model aprendre les característiques de les dades de manera efectiva.

10. Inicialització de punts de control (**–initial-checkpoint=/pretrained/cswin_tiny_224.pth** [3]): Utilitzar un model preentrenat ajuda a accelerar la convergència i millorar la precisió inicial del model, ja que els pesos del model ja han estat ajustats en una gran base de dades anterior.

Utilitzem aquests hiperparàmetres és aconseguir un model de classificació robust i precís capaç de gestionar múltiples tasques simultàniament. Això inclou la classificació d'imatges en diferents categories com aliments, begudes i ingredients. Els paràmetres seleccionats estan destinats a millorar la capacitat de generalització del model, reduir el sobreajustament i assegurar una convergència ràpida i estable.

[3] https://github.com/microsoft/CSWin-Transformer?tab=readme-ov-file

3.5 Resultats

Basat en els resultats proporcionats, aquí tens un resum de les conclusions sobre l'ús i no ús de **Normed-Softmax** i **Focal-loss** en el rendiment del model:

A partir de la taula, es pot observar l'impacte dels diferents mètodes en la presició del model:

1. **Model MDEH (ID 1)**:
    - Precisió general (top1_aggr): 78.843
    - Conclusió: El model MDEH base mostra un bon rendiment, però en alguns caps (com el de menjar) la precisió és relativament baixa.
2. **Model MDEH amb Focal-loss (ID 2)**:
    - Precisió general (top1_aggr): 80.077
    - Conclusió: L'ús de Focal-loss ha augmentat la precisió general del model, especialment en el cap de menjar.
3. **Model MDEH amb Normed-Softmax (ID 3)**:
    - Precisió general (top1_aggr): **81.13**
    - Conclusió: L'ús de Normed-Softmax ha aconseguit la precisió general més alta, amb millores en tots els caps, especialment en el cap de menjar.
4. **Model MDEH amb Normed-Softmax i Focal-loss (ID 4)**:
    - Precisió general (top1_aggr): 55.073
    - Conclusió: L'ús combinat de Normed-Softmax i Focal-loss ha reduït significativament el rendiment del model, indicant que pot haver-hi una interacció negativa entre aquestes dues tècniques, afectant el rendiment general del model.

A partir dels resultats anteriors, es pot concloure que l'ús de Normed-Softmax per separat (ID 3) millora significativament la precisió general del model i la precisió per cap, especialment en el non-food. No obstant, l'ús combinat de Normed-Softmax i Focal-loss (ID 4) redueix la presició del model, la qual cosa indica que aquesta combinació no és adequada per a aquest model. Per tant, tot i que aquestes tècniques tenen beneficis individuals, la seva combinació ha de ser considerada amb precaució.

ID	Methods	top1_aggr	top1_head_types	top1_head_ingredients	top1_head_food	top1_head_drinks
1	MDEH	78.843	96.987	96.451	80.226	99.131
2	MDEH Focal loss	80.077	97.053	96.625	81.419	99.227
3	**MDEH Normed-softmax**	**81.13**	**97.345**	**96.625**	**82.281**	**99.281**
4	MDEH Normed-softmax Focal loss	55.073	81.79	95.83	64.412	98.614

Figure 3.9: Resum de resultats

Per objectiu d'implementar fidelment el model proposat per Andrey, garantint que totes les tècniques de regularització, el tractament del desequilibri de dades i l'arquitectura multi-tasca es repliquin correctament.

Aquesta etapa assegurarà que el punt de partida del projecte estigui alineat amb els resultats ja obtinguts, proporcionant una base sòlida per a l'anàlisi posterior.

Quant als experiments de regularització, vam dur a terme quatre experiments i vam obtenir les següents conclusions: l'ús de Normed-Softmax per separat va aconseguir la precisió general més alta, amb millores en tots els heads, especialment en el non-food. Tanmateix, la combinació de Normed-Softmax i Focal-loss va reduir significativament la presició del model, la qual cosa indica que podria haver-hi una interacció negativa entre aquestes dues tècniques, afectant el rendiment general del model.

Malgrat haver identificat punts d'optimització i obtenir algunes conclusions rellevants sobre les tècniques de regularització, ens hem adonat que el nostre objectiu inicial, millorar el model proposat per Andrey, encara no s'ha assolit totalment. Aquesta trobada amb alguns obstacles ens ha mostrat la necessitat de continuar amb una anàlisi més profunda per comprendre millor la dinàmica entre les diferents tècniques de regularització i el seu impacte en la tasca multi-tasca que estem abordant.

Per tant, els treball futur inclouran una investigació més minuciosa sobre com millorar la integració de les tècniques de regularització seleccionades, com Normed-Softmax i Focal-loss, per optimitzar el rendiment global del model. A més, explorarem altres estratègies de regularització i abordarem més detalladament el desequilibri de dades per assegurar-nos que el model sigui robust i generalitzable.

A través d'aquesta continuació de la investigació, confiem en poder complir plenament amb l'objectiu inicial de millor el model, mentre millorem la seva presició en la tasca multi-task específica que estem abordant.

Capítol 4

Conclusions generals i treball futur

4.1 Conclusions

A la primera part, hem completat correctament tots els requisits del SDK per a iOS. En concret, hem implementat la recollida i el processament de la informació dels fotogrames, podent calcular amb precisió el punt central de l'escena i l'angle pel model d'estimació de volum i quantitat d'aliments, que es poden enviar a través de l'API al servidor. Mitjançant una anàlisi detallada dels requisits del sistema i un disseny integral, hem assegurat el rendiment i la usabilitat del SDK. Amb d'una sèrie d'algorismes, podem detectar i actualitzar dinàmicament el punt central de l'escena i calcular els angles en temps real, garantint la millor representació per estimació.

A la segona part, basant-nos en investigacions prèvies, hem aconseguit replicar la xarxa de reconeixement i hem obtingut uns resultats comparables als de les investigacions anteriors. Tot i que no vam poder optimitzar més el model, vam realitzar una exploració profunda en la tècnica de regularització, tractament del desequilibri de dades, i arquitectura multi-task. Els passos específics inclouen l'ús d'un conjunt de dades d'imatges de menjar de LogMeal per a l'entrenament, l'exploració de diferents arquitectures de model i estratègies d'optimització, i la selecció de els millors hiperparàmetres per a l'entrenament. Mitjançant l'avaluació en resultat d'anteriors, hem validat l'exactitud del model. Tot i que no vam millorar significativament el model, el nostre treball proporciona una base i referències per a futures investigacions d'optimització.

Combinant aquests dos enfocaments, aquest treball proporciona una solució integral per a la visualització i el reconeixement digital de menjar, fent contribucions importants al desenvolupament d'aplicacions mòbils i al reconeixement d'imatges basat en IA. La integració del SDK amb el model de reconeixement obre noves vies per a aplicacions en la identificació d'aliments, la gestió de la salut, l'experiència gastronòmica i la gestió de la dieta. El nostre treball no només demostra la viabilitat tècnica, sinó que també proporciona una base sòlida per a futures investigacions i aplicacions. A través del disseny de sistemes detallat i les proves completes, hem assegurat la fiabilitat i el valor pràctic de la solució. Aquesta solució integral té el potencial de tenir un impacte positiu en diversos camps, incloent recomanacions nutricionals personalitzades, la millora dels serveis en la indústria de la restauració i la transformació digital de la gestió de la salut.

4.2 Treball futur

Hi ha diverses direccions prometedores per a la continuació d'aquesta recerca. En primer lloc, es pot centrar en l'optimització la xarxa de reconeixement. Això pot incloure l'exploració de noves arquitectures de xarxes neuronals, es pot treballar en l'augment de la precisió del model mitjançant l'ampliació del conjunt de dades d'entrenament amb imatges de major qualitat i diversitat, així com la utilització de tècniques d'augmentació de dades.

En segon lloc, es pot millorar la usabilitat i l'eficiència del SDK. Això podria incloure l'optimització dels algorismes per a la detecció de punts centrals i el càlcul d'angles per reduir el temps de processament i augmentar la precisió. També es pot explorar la integració de funcionalitats addicionals, com ara la detecció automàtica d'aliments específics i la capacitat de reconèixer plats compostos per múltiples ingredients.

A més, una altra línia de treball futur podria ser la integració del SDK amb altres tecnologies emergents. Per exemple, es podria explorar la utilització de la realitat augmentada (AR) per millorar la visualització dels models 3D de menjar en temps real. Això permetria als usuaris veure una representació digital del menjar directament al seu entorn físic, oferint una experiència més immersiva i interactiva.

Finalment, es pot treballar en l'aplicació pràctica de la solució desenvolupada en diferents àmbits. Això inclou la col·laboració amb professionals de la salut per a desenvolupar aplicacions de gestió de dietes personalitzades basades en la identificació d'aliments. També es pot explorar la integració amb sistemes de restaurants per a millorar els serveis de menús digitals i la gestió de comandes, així com amb aplicacions de comerç electrònic per a proporcionar descripcions de productes alimentaris més detallades i visuals.

En resum, les futures investigacions poden centrar-se en l'optimització tècnica, la millora de l'experiència d'usuari, la integració amb noves tecnologies i l'aplicació pràctica en diferents sectors, amb l'objectiu final de proporcionar solucions més eficaces, precises i accessibles per a la visualització i el reconeixement digital de menjar.

Bibliography

[1] Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". Advances in Neural Information Processing Systems (NIPS), 25, pp. 1097-1105.

[2] Hypertension, Newsroom Factsheets Detail Hypertension. WHO. 2023.

[3] Mental disorders, Newsroom Factsheets Detail Mentaldisorders. WHO. 2022.

[4] ARKit, developer.apple.com/documentation/arkit. developer, Apple, 2023.

[5] Azarcoya-Cabiedes, Willy. Centre òptic. ResearchGate. (2014).

[6] Baeldung (2023). Principal point. baeldung.

[7] Vuforia (2023). ?Spatial Frame of Reference for Unity? .

[8] Bhalaji Nagarajan Rupali Khatun, Marc Bolaños Eduardo Aguilar Leonardo Angelini Mira El Kamali Elena Mugellini Omar Abou Khaled Noemi Boqué Lucia Tarro Petia Radeva (2021). ?Nutritional Monitoring in Older People Prevention Services?.

[9] Ankit Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby (2020). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". arXiv preprint arXiv:2010.11929.

[10] Núñez Picado, Andrey. Develop a deep learning model for food image classification, UPC-UB-URV, 2024.

[11] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, Baining Guo. CSWin Transformer: A General Vision Transformer Backbone with Cross-Shaped Windows. arXiv:2107.00652, 2021.

[12] Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems (pp. 1097-1105).

[13] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is All You Need. In Advances in Neural Information Processing Systems (pp. 5998-6008).

[14] Parikh, A., Täckström, O., Das, D., Uszkoreit, J. (2016). A Decomposable Attention Model for Natural Language Inference. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 2249-2255).

[15] Posner, M. I., Petersen, S. E. (1990). The Attention System of the Human Brain. Annual Review of Neuroscience, 13, 25-42.

[16] Li, Y., Bai, Y., Zhang, W., Wang, J., Yang, X. (2021). Enhanced Local Attention for Transformer. Proceedings of the 2021 Conference on Neural Information Processing Systems (NeurIPS 2021).

[17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is All You Need. In Advances in Neural Information Processing Systems (pp. 5998-6008).

[18] Hornik, K., Stinchcombe, M., White, H. (1991). Multilayer feedforward networks are universal approximators. Neural Networks, 4(2), 359-366.

[19] Hendrycks, D., Gimpel, K. (2016). Gaussian Error Linear Units (GELUs). arXiv preprint arXiv:1606.08415.

[20] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929-1958.

[21] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems (pp. 8026–8037).

[22] Lin, T. Y., Goyal, P., Girshick, R., He, K., Dollár, P. (2017). Focal Loss for Dense Object Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(2), 318-327.

[23] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks, 2018.

[24] Loshchilov, I., Hutter, F. (2019). Decoupled Weight Decay Regularization. In Proceedings of the Seventh International Conference on Learning Representations.

[25] Kingma, D. P., Ba, J. (2014). Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980.

[26] Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., Yoo, Y. (2019). CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features. In Proceedings of the IEEE International Conference on Computer Vision (pp. 6023-6032).

Appendix

Figure 1: Mostres d'imatges en el conjunt de begudes al base de dade de LogMeal

Figure 2: Mostres d'imatges en el conjunt de menjar al base de dade de LogMeal

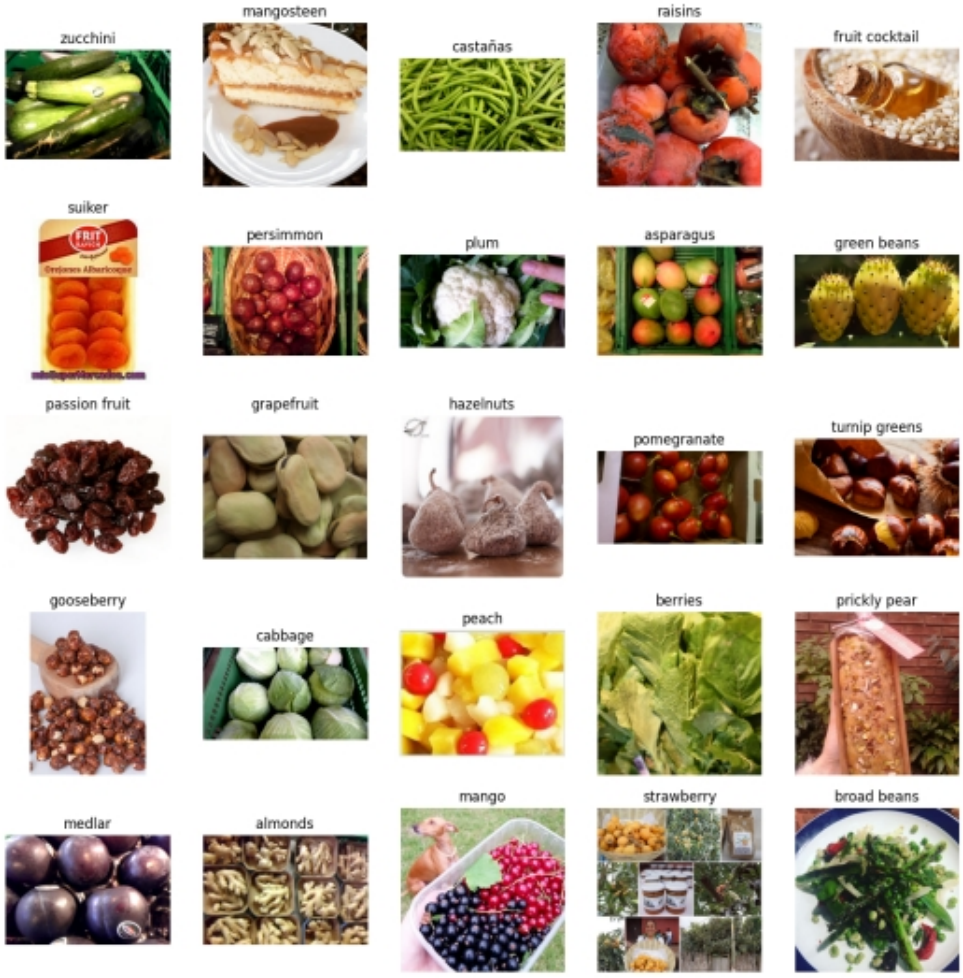

Figure 3: Mostres d'imatges en el conjunt d'ingredients al base de dade de LogMeal

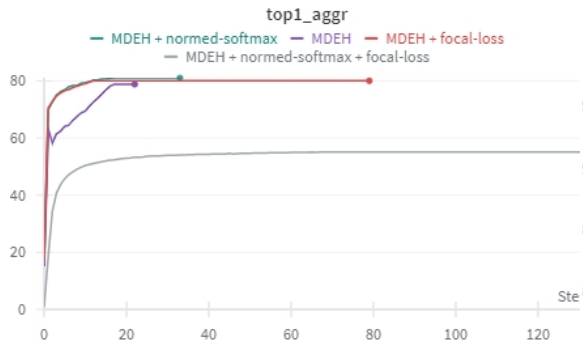

Figure 4: top1 aggr

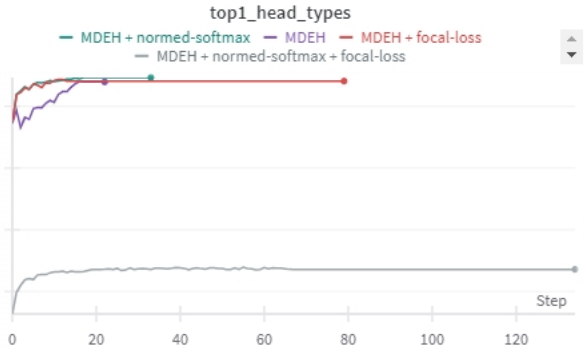

Figure 5: top1 head types

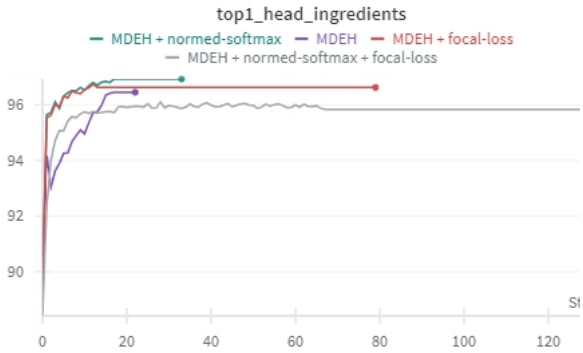

Figure 6: top1 head ingredients

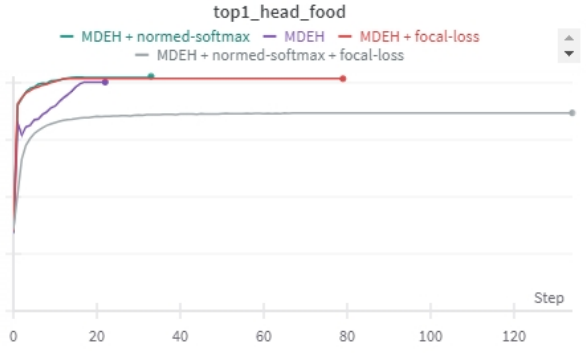

Figure 7: top1 head food

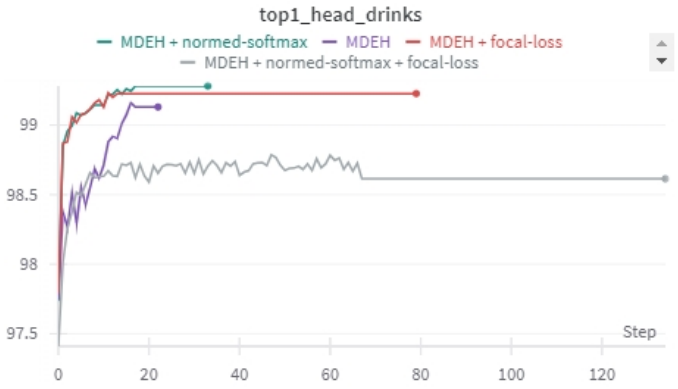

Figure 8: top1 head drinks

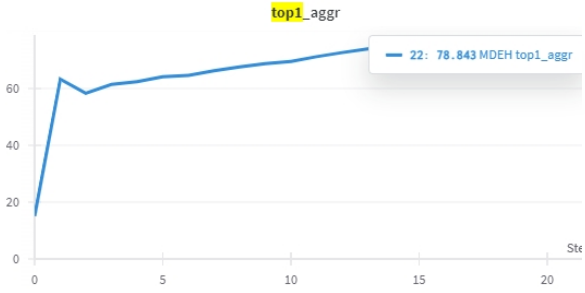

Figure 9: MDEH top1 aggr

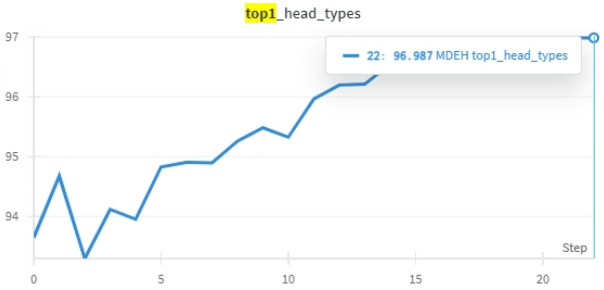

Figure 10: MDEH top1 head types

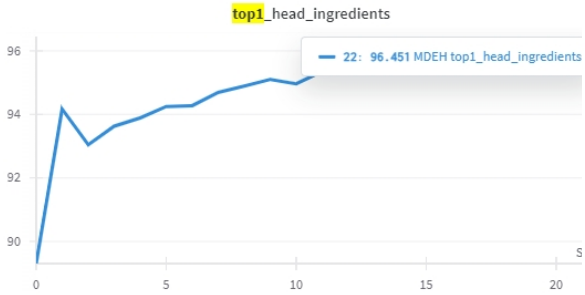

Figure 11: MDEH top1 head ingredients

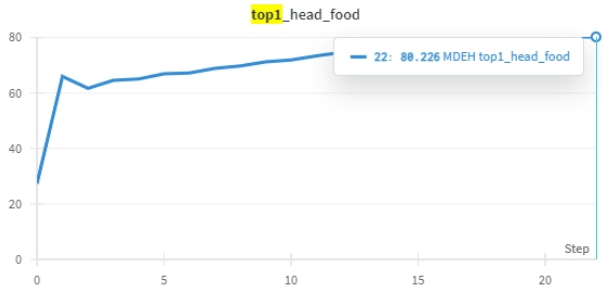

Figure 12: MDEH top1 head food

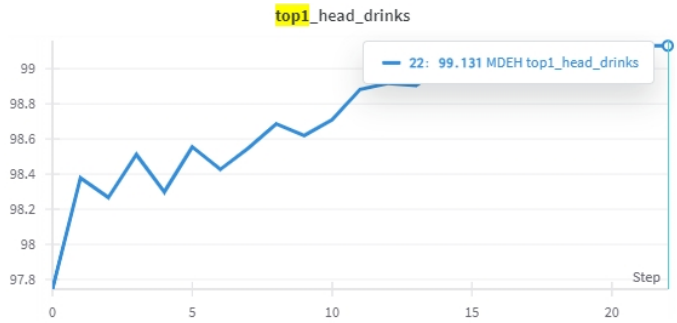

Figure 13: MDEH top1 head drinks

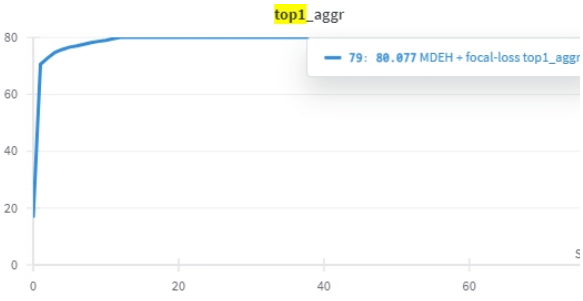

Figure 14: MDEH + Focal loss top1 aggr

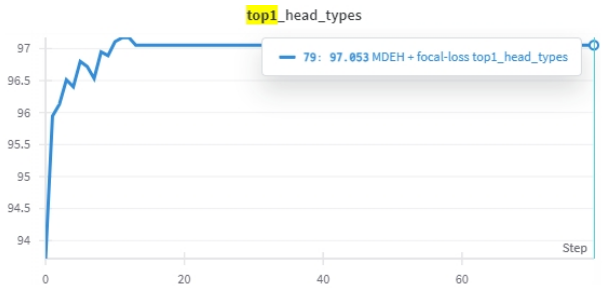

Figure 15: MDEH + Focal loss top1 head types

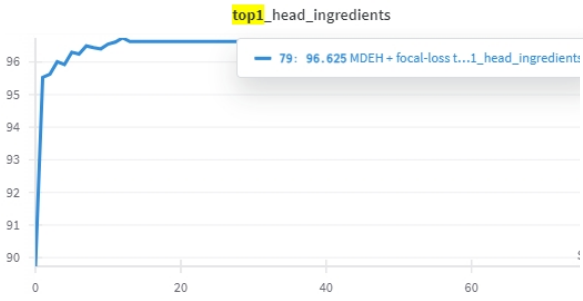

Figure 16: MDEH + Focal loss top1 head ingredients

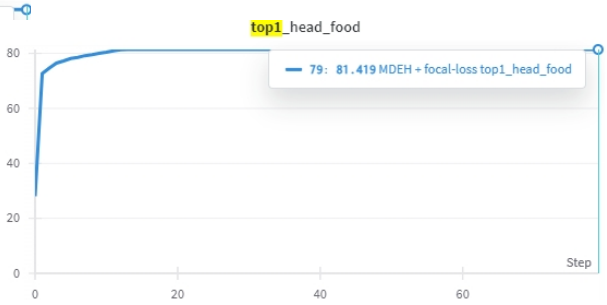

Figure 17: MDEH + Focal loss top1 head food

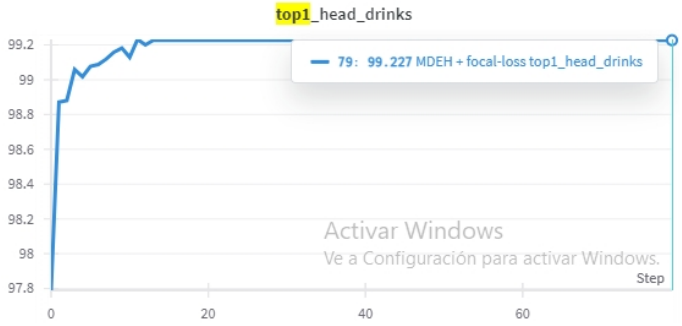

Figure 18: MDEH + Focal loss top1 head drinks

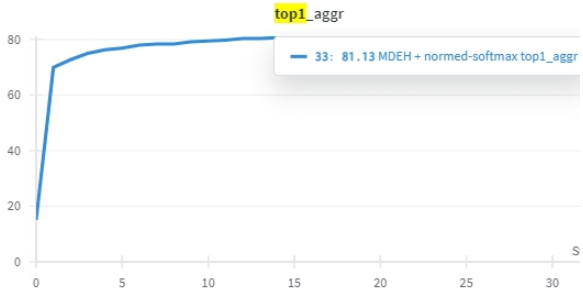

Figure 19: MDEH + Normed-softmax top1 aggr

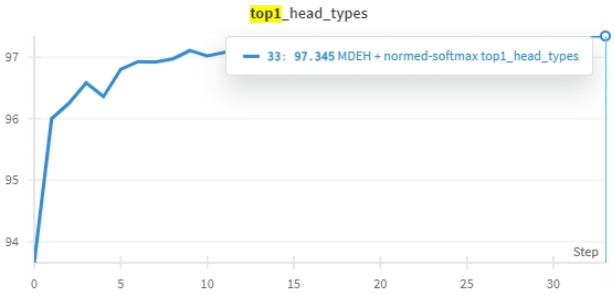

Figure 20: MDEH + Normed-softmax top1 head types

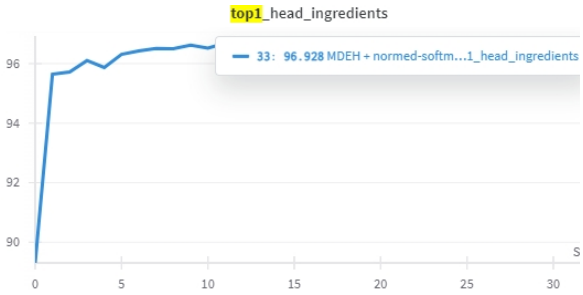

Figure 21: MDEH + Normed-softmax top1 head ingredients

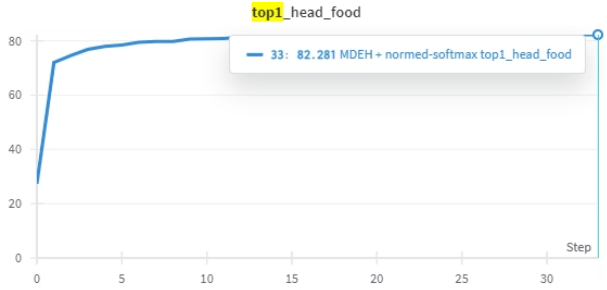

Figure 22: MDEH +Normed-softmax top1 head food

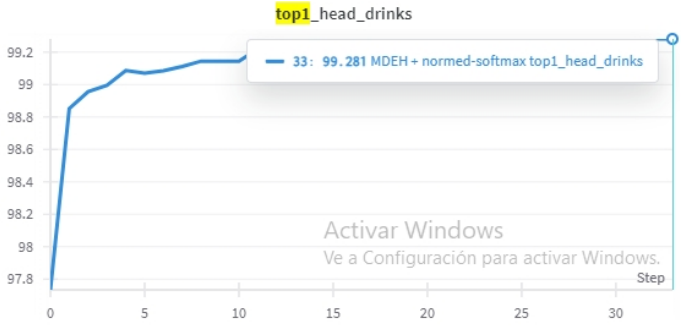

Figure 23: MDEH + Normed-softmax top1 head drinks

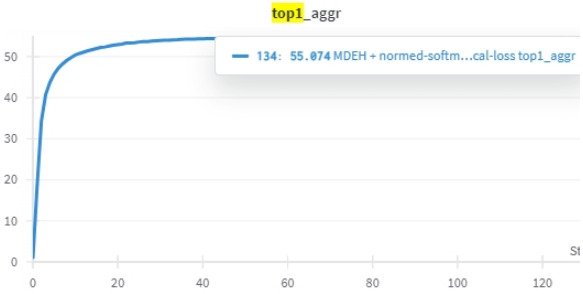

Figure 24: MDEH + Normed-softmax + Focal loss top1 aggr

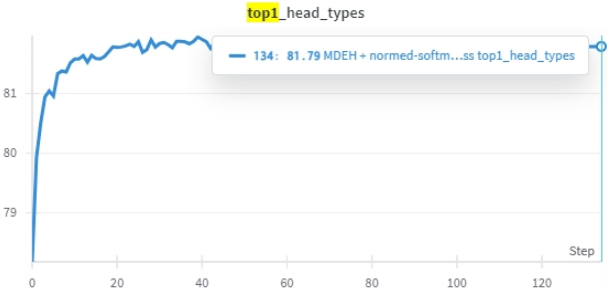

Figure 25: MDEH + Normed-softmax + Focal loss top1 head types

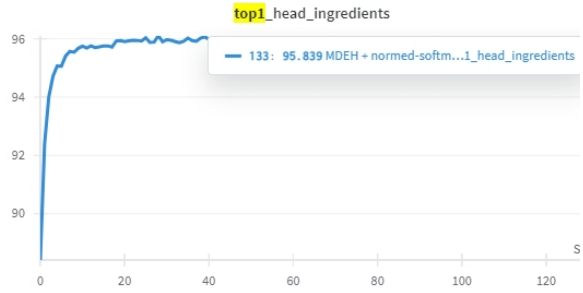

Figure 26: MDEH + Normed-softmax + Focal loss top1 head ingredients

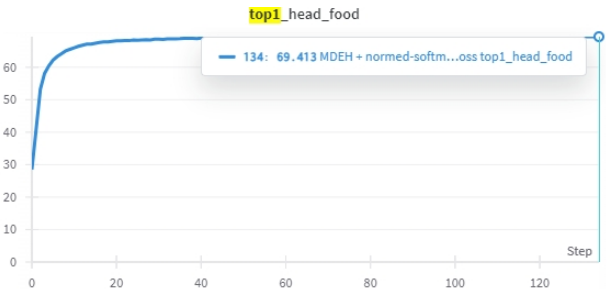

Figure 27: MDEH +Normed-softmax + Focal loss top1 head food

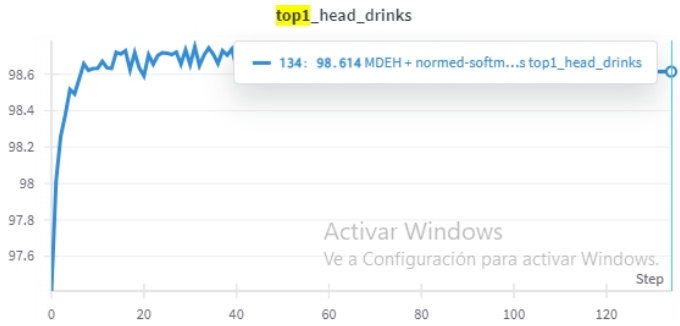

Figure 28: MDEH + Normed-softmax + Focal loss top1 head drinks