

UNIVERSITÀ POLITECNICA DELLE MARCHE
FACOLTÀ DI INGEGNERIA
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione



TESI DI LAUREA

Progettazione e implementazione di un sistema per il riconoscimento e il tracciamento di prodotti vitivinicoli di qualità

Design and implementation of a system for recognition and tracking of quality wine products

Relatore

Prof. Domenico Ursino

Candidato

Compagnoni Paolo

Correlatore

Dott. Ferrante Marco

*Una nave in porto è al sicuro,
ma non è per questo che le navi sono state costruite*

Benazir Bhutto

Sommario

Negli ultimi anni la Customer Analytics ha assunto un'importanza sempre maggiore, in quanto, riuscendo a capire i comportamenti dei clienti e le loro preferenze, permette di prendere decisioni strategiche e tattiche. Le promozioni giocano un ruolo importantissimo nel soddisfare ed attirare nuovi clienti. In questa tesi sono stati analizzati i dati relativi alle promozioni ed alle vendite dell'azienda Fater. In particolare è stata realizzata una fase di ETL dei dati, seguita da un'analisi descrittiva. Successivamente è stata effettuata una fase di clustering, seguita da una di classificazione, per valutare quanto le promozioni abbiano influenzato le vendite.

Keyword: Data Analytics, Customer Analytics, Promozioni, Extract Transform and Load, Exploratory Data Analysis, Cluster, Classificazione, Pandas

Introduzione	1
1 L'agricoltura di precisione	3
1.1 Introduzione all'agricoltura di precisione	3
1.2 Storia dell'agricoltura di precisione	3
1.3 Machine Learning e Deep Learning nell'agricoltura di precisione	6
1.3.1 Introduzione	6
1.3.2 Il Machine Learning	7
1.3.3 Algoritmi Machine Learning	8
1.3.4 Applicazioni del machine learning in agricoltura	9
1.3.5 Il Deep Learning	12
1.3.6 Vantaggi e svantaggi in agricoltura	16
1.3.7 IoT in agricoltura	16
1.3.8 Applicazioni dell'IoT in agricoltura	18
2 Il tracciamento dei prodotti	19
2.1 Introduzione	19
2.1.1 Responsabilità per il tracciamento dei prodotti	20
2.1.2 Il tracciamento dei prodotti comparato agli altri settori	21
2.1.3 Le motivazioni allo sviluppo	22
2.1.4 Sviluppi futuri	22
2.2 Campi di utilizzo	23
2.2.1 Salute pubblica	23
2.2.2 Industria	27
2.2.3 Tracciabilità dei prodotti sfusi	30
2.2.4 Tracciabilità della carne e pollame	32
2.3 La blockchain nella tracciabilità dei prodotti	33
2.3.1 Introduzione alla blockchain	33
2.3.2 Casi d'uso nella tracciabilità	34
2.3.3 Configurazioni della blockchain	35
3 Descrizione generale del progetto	37
3.1 Premessa	37
3.2 Scopo generale dell'applicazione	37
3.3 Strumenti software utilizzati	38

3.3.1	Unity	38
3.3.2	Vuforia Engine	39
3.3.3	Polycam	40
3.3.4	Blender	41
3.3.5	Vuforia Model Target Generator	42
3.3.6	MongoDB	42
3.3.7	Google Earth Studio	44
3.3.8	OpenWeather	45
3.4	Strumentazione IoT utilizzata	46
3.5	Funzionamento dell'applicazione	47
4	Progettazione del sistema	52
4.1	Scelte progettuali effettuate	52
4.2	Diagrammi delle classi	53
4.2.1	La classe IotAPICaller	53
4.2.2	La classe OpenWeatherAPICaller	55
4.2.3	La classe ScriptManager	59
4.2.4	La classe MongoDBConnector	60
4.3	Sequence Diagram	61
4.3.1	Introduzione	61
4.3.2	Sequence Diagram del progetto	61
4.4	Animazioni in Unity	63
4.4.1	Introduzione	63
4.4.2	Le animazioni utilizzate nel progetto	63
4.5	Il database MongoDB	64
5	Implementazione e manuale utente	66
5.1	Creazione ed importazione di un modello tridimensionale di un prodotto	66
5.1.1	Creazione del modello tridimensionale dell'oggetto tramite smartphone	66
5.1.2	Raffinamento del modello tridimensionale con Blender	67
5.1.3	Creazione ed importazione di un modello in Vuforia Engine	67
5.2	Implementazione delle classi principali	68
5.2.1	Implementazione della classe IotAPICaller	68
5.2.2	Implementazione della classe OpenWeatherAPICaller	71
5.2.3	Implementazione della classe ScriptManager	75
5.2.4	Implementazione della classe MongoDBConnector	79
5.3	Implementazione delle animazioni in Unity	81
5.3.1	Menù principale	81
5.3.2	Selezione dell'annata	83
5.4	Manuale utente	83
5.4.1	Premessa	83
5.4.2	Effettuare il riconoscimento di un prodotto	83
5.4.3	Utilizzo delle sezioni principali	83
6	Confronto con sistemi di tracciamento	85
6.1	Aspetti in comune del progetto con i sistemi di tracciamento	85
6.2	Aspetti differenti del progetto con i sistemi di tracciamento	85

7 Discussione in merito al lavoro svolto	86
7.1 Confronto con l'app Vivino	86
7.2 SWOT Analysis	86
7.3 Sviluppi futuri	86
8 Conclusioni	87
Bibliografia	88
Ringraziamenti	89

Elenco delle figure

3.1	Logo di Unity	38
3.2	Logo di Vuforia Engine	39
3.3	Logo di Polycam	40
3.4	Logo di Blender	41
3.5	Logo di MongoDB	42
3.6	Logo di OpenWeather	45
3.7	La stazione IoT presente nel campo dell'azienda vitivinicola Strappelli	46
3.8	Sensore di temperatura ed umidità della foglia	47
3.9	Menu principale dell'applicazione	47
3.10	Pulsante che chiede all'utente di iniziare il riconoscimento del prodotto vitivinicolo	48
3.11	Scena AR dedicata al riconoscimento del prodotto vitivinicolo	49
3.12	Finestra per la selezione dell'annata del prodotto vitivinicolo	50
3.13	Screenshot della sezione Osserva	50
3.14	Screenshot della sezione Ascolta	51
3.15	Screenshot della sezione Racconta	51
4.1	Class Diagram della classe <i>IotAPICaller</i>	55
4.2	Class Diagram della classe <i>OpenWeatherAPICaller</i>	58
4.3	Class Diagram della classe <i>ScriptManager</i>	59
4.4	Class Diagram della classe <i>MongoDBConnector</i>	61
4.5	Sequence Diagram del flusso di esecuzione principale dell'applicazione	62
4.6	La struttura dell'Animator controller <i>Menu Controller</i>	64
4.7	La struttura dell'Animator controller "Scene2IntroController"	64
4.8	Documento MongoDB della cantina Strappelli	65
4.9	Struttura del campo <i>ScriptSommelier</i>	65
5.1	Il set utilizzato per creare il modello 3D	67
5.2	Modello 3D prima dell'utilizzo di Blender	68
5.3	Modello 3D dopo l'utilizzo di Blender	69
5.4	Posizionamento dell'oggetto <i>ModelTarget</i> nella scena	69
5.5	Sequenza di operazioni svolte all'avvio dell'applicazione dall'animazione <i>MenuFadeIn</i>	82
5.6	Sequenza di operazioni svolte dall'animazione <i>ChoiceOsserva</i>	82

Elenco delle tabelle

Introduzione

Negli ultimi decenni i dati hanno assunto un'importanza sempre maggiore nella vita di un'azienda e, soprattutto, rappresentano un vantaggio competitivo, se sono ben utilizzati. Per fare ciò, essi devono essere analizzati e studiati a fondo. L'analisi dei dati è un processo di ispezione, pulizia, trasformazione e modellazione di dati con il fine di evidenziare informazioni utili che supportino le decisioni strategiche aziendali. Se le decisioni aziendali vengono prese senza l'utilizzo della conoscenza dei dati, non parliamo di una *decisione strategica* ma di una semplice *ipotesi*.

Le aziende sono in possesso di una grande mole di dati; esse, spesso, non riescono ad interpretarli e sfruttare l'enorme potenziale che essi hanno. Per capire questo concetto, riportiamo una frase del matematico francese, Henri Poincaré, ovvero: "la scienza è fatta di dati come una casa è fatta di pietre. Ma un ammasso di dati non è scienza più di quanto un mucchio di pietre sia una vera casa". Questa frase ribadisce il fatto che, nonostante un'azienda sia in possesso di tantissimi dati, non necessariamente questi portano informazioni utili all'azienda. Tuttavia, attraverso una fase accurata di analisi, possono portare l'azienda stessa al successo.

I dati hanno un potenziale informativo enorme, che può aiutare le imprese sia a conoscere meglio se stesse (utilizzando i dati interni) sia il proprio mercato (utilizzando dati esterni), ma soprattutto i propri clienti. Secondo Peter Drucker, considerato il padre del management moderno, lo scopo di un'azienda è quello di creare e mantenere stretto un cliente attraverso il marketing e l'innovazione. È molto importante soddisfare il cliente con i propri prodotti/servizi; però, la maggior parte delle volte questo è molto difficile a causa della concorrenza. Per questo motivo, nell'ultimo periodo, sono tante le imprese che hanno iniziato a considerare di fondamentale importanza la *customer analytics*, per conoscere i propri clienti ed ottenere risultati migliori. Grazie a questa attività si riescono a capire i comportamenti dei propri clienti e le preferenze di questi ultimi al fine di prendere decisioni commerciali strategiche e tattiche.

In generale si può dire che tutti i dati sono importanti per il business, perché la maggior parte aiuta a scoprire e analizzare le aspettative dei clienti e le opportunità di crescita più efficaci. I dati utilizzati per eseguire questo tipo di analisi, la maggior parte delle volte, possono provenire da diversi reparti di un'azienda e riguardano le vendite, gli acquisti, i prodotti, ma anche le promozioni realizzate dall'azienda stessa.

Queste ultime svolgono un ruolo fondamentale nel soddisfare e attirare nuovi clienti, in quanto stimolano la richiesta del prodotto/servizio che, all'occhio del cliente, deve sembrare più attraente e innovativo. In base agli obiettivi dell'azienda, le strategie promozionali possono essere di tipo *pull* o *push*. Le prime cercano di far sì che i clienti "tirino" i prodotti

dall'azienda, attraverso, per esempio, l'utilizzo di sconti. Le seconde cercano di spingere il prodotto dall'azienda verso distributori e rivenditori. Le prime permettono di costruire un legame più forte con il cliente, in quanto è a stretto contatto con questo.

In questo elaborato sono stati analizzati i dati relativi alle promozioni e alle vendite dell'azienda Fater. In una prima fase viene effettuata una pulizia dei dati; successivamente, viene eseguita un'esplorazione di questi per comprenderli meglio. Ci sarà, anche, un'analisi descrittiva dei contenuti più interessanti all'interno dei dataset. L'obiettivo finale è quello di capire quanto le promozioni, realizzate dall'azienda, hanno influenzato le vendite; per fare ciò, viene eseguita una fase di clustering; successivamente, per riuscire a esprimere quanto un nuovo prodotto in promozione influenzerà la vendita, viene svolta una fase di classificazione.

La presente tesi è composta da sette capitoli strutturati come di seguito specificato:

- Nel Capitolo 1 saranno introdotti il mondo della customer analytics e, successivamente, il ruolo delle promozioni nelle campagne di marketing.
- Nel Capitolo 2 si presenteranno le attività relative alle campagne promozionali in Fater. In particolare, si introduciranno le due tipologie di sconti utilizzate e la gerarchia dei clienti.
- Nel Capitolo 3 verrà introdotto il sistema dell'azienda riguardante le promozioni; successivamente, saranno illustrati i set di dati relativi alle promozioni e alle vendite dell'azienda.
- Nel Capitolo 4 verranno analizzate le fasi di ETL e di EDA sui dataset delle promozioni e delle vendite e sarà effettuata un'analisi descrittiva dei dati.
- Nel Capitolo 5 verranno analizzate le attività di clustering e classificazione dei dataset.
- Nel Capitolo 6 saranno riportati i risultati ottenuti.
- Nel Capitolo 7 verranno tratte le conclusioni e verranno delineati alcuni possibili sviluppi futuri.

CAPITOLO 1

L'agricoltura di precisione

Nel primo capitolo, si introduce il concetto di agricoltura di precisione, fornendo inizialmente una breve definizione. Successivamente, si esplorerà il percorso storico che condurrà dalle origini dell'agricoltura fino ai giorni nostri, dove l'agricoltura di precisione è diventata una pratica diffusa. Questo permetterà di comprendere l'evoluzione della ricerca in questo campo fino all'attuale stato dell'arte.

In seguito, si esamineranno le due tecnologie fondamentali su cui si basa l'agricoltura di precisione: il machine learning e il deep learning. Per entrambe queste tecnologie, si presenterà in modo conciso le tecniche più utilizzate, sia in generale che nel contesto specifico dell'agricoltura di precisione, mettendone in luce vantaggi e svantaggi.

Infine, nell'ultima sezione, affronteremo il tema dell'Internet delle cose (IoT), che rappresenta il contesto in cui si colloca l'applicazione sviluppata in Unity.

1.1 Introduzione all'agricoltura di precisione

L'agricoltura di precisione può essere definita come un approccio integrato all'agricoltura che impiega dati geo-referenziati, tecnologie digitali, sensori, automazione e analisi avanzate per monitorare, gestire e ottimizzare con precisione ogni aspetto del processo agricolo, dalla preparazione del suolo alla coltivazione, dalla raccolta alla distribuzione.

L'obiettivo principale è massimizzare l'efficienza nell'uso delle risorse, minimizzare gli impatti ambientali, migliorare la produttività e la qualità delle colture, e garantire una gestione sostenibile delle aziende agricole. L'agricoltura di precisione si basa su un approccio personalizzato, in cui le decisioni vengono prese in tempo reale in base alle specifiche condizioni e necessità di ogni singola area coltivata, consentendo agli agricoltori di ottimizzare i rendimenti e la redditività, riducendo, al contempo, gli sprechi e l'impatto ambientale.

1.2 Storia dell'agricoltura di precisione

Gli antichi, pur comprendendo le differenze nella produzione agricola tra i vari campi, sembravano dare scarsa importanza alle variazioni all'interno di ciascun campo, come testimonia Catone nel 160 a.C.

I Romani valutavano l'acquisto di terra principalmente in base alla loro impressione generale sulla gestione della fattoria, la sua posizione nel paesaggio e le caratteristiche del suolo. Per fertilizzare i campi, utilizzavano una serie di metodi tra cui concimi organici, compost e il

liquido residuo dall’estrazione dell’olio d’oliva. Tuttavia, a causa del pesante carico di lavoro richiesto per le pratiche agricole fondamentali, sembra che abbiano dato scarsa attenzione alla variazione all’interno dei singoli campi. I proprietari terrieri si concentravano di più sulla gestione degli schiavi o dei liberti che sulla differenziazione dei terreni. Inoltre, l’acquisizione di nuove terre era spesso più prioritaria della gestione delle aree degradate all’interno dei campi.

Nei secoli successivi, gli scienziati iniziarono a definire gli obiettivi della scuola agraria. Il primo tema sperimentale che emerse fu la necessità di affrontare la variabilità nella resa delle colture dovuta all’eterogeneità dei suoli. Nonostante gli sforzi per individuare aree il più omogenee possibile per condurre esperimenti sul campo, l’ostacolo dell’eterogeneità del terreno continuava a sfidare i ricercatori.

Negli anni venti del secolo scorso, si compirono notevoli progressi per migliorare le decisioni legate alla variabilità spaziale su piccola scala. Robert A. Fisher iniziò un lavoro rivoluzionario presso la Rothamsted Experiment Station di Harpenden, in Inghilterra, nel 1919. In sette anni, sviluppò una serie di strumenti statistici che sarebbero diventati la base per la maggior parte degli esperimenti su piccole aree, e persino interi campi. L’applicazione dei principi enunciati da Fisher e l’ampliamento degli strumenti statistici per affrontare le sfide legate alle pendenze e alle differenze sistematiche nei terreni, si rivelarono fondamentali per mitigare l’impatto della variabilità spaziale negli esperimenti su piccole aree condotti da generazioni di ricercatori sul campo. Tuttavia, nessuno di questi strumenti si è dimostrato particolarmente idoneo a gestire la variabilità nei fattori quali nutrienti, infestazioni da erbe infestanti, presenza di insetti, dosaggi di semina o altri input gestionali nei campi.

A partire dal 1920, gran parte dell’attenzione nell’agricoltura orientata alla specificità del suolo si è concentrata sulla gestione dei nutrienti per le coltivazioni. L’analisi del terreno è emersa come un aspetto cruciale, poiché è stata riconosciuta come un mezzo per valutare la capacità del suolo di fornire nutrienti, un concetto che risale ai lavori di Sprengel nel 1839.

Nella procedura di analisi del terreno, un campione di suolo viene prelevato dal campo per poi essere sottoposto ad un trattamento per rimuovere detriti e piccoli sassi. Successivamente, il campione viene mescolato con un estrattore liquido, filtrato e il contenuto del nutriente di interesse viene quantificato per consentire un confronto con una quantità standard correlata a una potenziale risposta della coltura. Questa risposta può risultare vantaggiosa per la coltivazione o, in alcuni casi, avere effetti tossici, a seconda del tipo di elemento o composto estratto e della sua quantità nel terreno.

La prima raccomandazione conosciuta per il campionamento del suolo al fine di gestire la variabilità nei campi agricoli fu pubblicata da Linsley e Bauer nel 1929. Questa raccomandazione suggeriva agli agricoltori di prelevare campioni di suolo a una profondità di 15 cm e analizzarli seguendo una griglia di 0,4 ettari, con ulteriori prelievi a 30 cm di profondità. La motivazione dietro questa pratica era legata alla difficoltà di diffondere il calcare agricolo in campi con terreno acido.

Nel 1938, gli agricoltori avevano ormai a disposizione numerose macchine per l’applicazione di fertilizzanti, molte delle quali venivano utilizzate regolarmente, comprese quelle per la distribuzione, la localizzazione in collina e l’applicazione vicino ai semi.

Dal decennio degli anni ’50 fino ai giorni nostri, il campionamento base del suolo è stato principalmente rappresentato da campioni composti che rappresentano un intero campo. Nonostante i ricercatori esperti nella variazione spaziale dei nutrienti delle colture abbiano consigliato di includere solo terreni relativamente omogenei e simili in un campione composito, essi venivano prelevati da diverse zone all’interno dei campi, delimitate dai confini imposti dagli agricoltori, considerando la variabilità dei suoli al loro interno.

Negli anni ’60, si verificarono anche importanti avanzamenti nell’affrontare direttamente la variabilità spaziale dei nutrienti del suolo, con l’introduzione del campo statistico della

geo-statistica da parte di un ricercatore canadese di nome Matheron nel 1963.

Poiché il campionamento del suolo o di qualsiasi altra entità all’interno di un campo agricolo identifica soltanto la piccola area da cui vengono prelevati campioni, piante, parti di piante o misurazioni, la stragrande maggioranza delle zone all’interno del campo rimane sconosciuta rispetto ai valori osservati. Di conseguenza, è necessario stimare o *interpolare* i valori delle aree non campionate nel campo per prendere decisioni basate sui risultati del campionamento. Il *Kriging* è il metodo preferito per raggiungere questo obiettivo, sebbene richieda un insieme minimo di almeno 30 osservazioni per essere applicato con successo.

Negli anni ’80, il Dipartimento della Difesa degli Stati Uniti ottenne finanziamenti dal Congresso per sviluppare il sistema di posizionamento satellitare noto come *GPS*. La rete GPS era composta da 24 satelliti e fu completata nel 1994. L’anno precedente, nel 1993, venne siglato un accordo tra il Dipartimento della Difesa e il Dipartimento dei Trasporti, che consentì l’utilizzo civile del sistema GPS. Questa apertura del GPS al settore agricolo rappresentò una svolta fondamentale per l’agricoltura di precisione, e in breve tempo diverse aziende iniziarono a offrire sistemi GPS per l’agricoltura.

Inizialmente, i sistemi GPS originali avevano una precisione che non permetteva la localizzazione entro pochi metri dalla posizione desiderata. Tuttavia, le aziende agricole e gli agricoltori risolsero questo problema installando torri di correzione cinetica in tempo reale, che migliorarono notevolmente la precisione del posizionamento. Va notato che i segnali GPS provenienti direttamente dai satelliti potevano presentare errori dovuti alle condizioni atmosferiche. Per affrontare questi errori, vennero sviluppati ricevitori e trasmettitori differenziali satellitari correttivi, come quelli adottati nel sistema *GreenStar* di John Deere.

La determinazione della topografia può presentare un impatto significativo sullo sviluppo del suolo e sul movimento dell’acqua all’interno del terreno, influenzando così la produttività delle coltivazioni. La misurazione dell’elevazione può essere effettuata con il GPS differenziale dal quale è possibile ottenere non solo la latitudine e la longitudine, ma anche l’altitudine della posizione GPS.

Un approccio innovativo al rilevamento topografico è stato introdotto con lo sviluppo del *LiDAR* (*Light Detecting And Ranging*). Il LiDAR ha visto la luce poco dopo l’invenzione del laser, agli inizi degli anni ’60.

Oltre alla topografia, diversi altri strumenti sono stati impiegati nello sviluppo delle zone di gestione dei nutrienti. Tra questi strumenti, le immagini satellitari hanno svolto un ruolo cruciale, soprattutto per quanto riguarda la gestione dell’azoto. Con il tempo, le immagini satellitari si sono rivelate strumenti preziosi per delimitare le zone di gestione dell’azoto durante la stagione agricola, integrando dati ottenuti tramite campionamento del suolo e altri metodi. Successivamente, sono stati sviluppati sensori basati sulle proprietà di trasmittanza elettrica del suolo ampiamente utilizzati per la delineazione delle zone di gestione dei nutrienti.

La conducibilità elettrica è stata anche direttamente correlata ai livelli di nitrato nel suolo in terreni che altrimenti sembravano uniformi. Inoltre, sia la conducibilità elettrica che il magnetismo possono essere utilizzati per rilevare differenze nella capacità di ritenzione dell’acqua, nei contenuti idrici del suolo, nella capacità di scambio cationico, nella porosità, nella salinità e nei gradienti di temperatura. Tuttavia, in molti campi più di un fattore varia in modo indipendente dagli altri. In campi *multivariabili*, i sensori di conducibilità elettrica e di flusso magnetico sono strumenti efficaci per identificare pattern e suddividere le zone, spesso superando altre tecniche.

La costruzione di algoritmi finalizzati all’applicazione dell’azoto durante la stagione si basa sulla previsione delle rese da parte del sensore in una striscia di azoto, confrontata con altre aree all’interno del campo. Il risultato di questo lavoro è stata la commercializzazione del sensore attivo-ottico *GreenSeeker* nel 2002. Questo applicatore aveva la capacità di operare

alle velocità del trattore nel campo e applicare l’azoto a ogni metro quadro di coltura in modo indipendente grazie alla sua serie di sensori e gruppi di ugelli posizionati ad ogni metro di larghezza della barra di spruzzatura.

La commercializzazione di successo dei sensori del suolo correlati ai nutrienti include il sensore del pH del suolo Veris Technologies. In uno studio, l’uso del sensore di pH Veris ha correlato il pH mappato dal sensore con i veri modelli di pH del terreno rispetto alle tecniche standard di campionamento del suolo site-specific, riducendo gli errori nelle raccomandazioni di dosaggio della calce del 50%. Veris commercializza anche l’*Optic Mapper*, che utilizza un rilevatore di infrarossi vicino all’interno del suolo per stimare il contenuto di materia organica di quest’ultimo.

1.3 Machine Learning e Deep Learning nell’agricoltura di precisione

1.3.1 Introduzione

L’agricoltura costituisce il pilastro fondamentale della nostra economia in quanto la crescente domanda di prodotti alimentari, derivante dall’incremento della popolazione, continua a crescere in modo costante. Per rispondere efficacemente a questa esigenza, il settore agricolo deve compiere notevoli progressi, ad esempio nell’effettuare calcoli precisi sulla produzione, sfruttando le attrezzature agricole più avanzate e aggiornate disponibili sul mercato. L’obiettivo principale di tali progressi è quello di soddisfare la crescente richiesta di coltivazioni alimentari di alta qualità.

Per avviare il processo verso l’agricoltura di precisione, si compiono passi iniziali fondamentali tra cui la previsione meteorologica e la valutazione degli effetti dei vari fertilizzanti, sfruttando il telerilevamento e i sensori per monitorare lo stato di salute delle colture.

Un importante passo avanti nell’evoluzione dell’agribusiness è consistito nell’introduzione dei sensori wireless nell’agricoltura e nell’industria alimentare. Questa innovazione ha aperto nuove possibilità nel monitoraggio e nell’ottimizzazione delle attività agricole e industriali legate al settore alimentare, contribuendo, così, a migliorare l’efficienza e la qualità della produzione agroalimentare.

Il concetto di gestione agricola si basa sulla raccolta di dati mediante osservazioni e misurazioni, nonché sulla capacità di rispondere in modo mirato alla variabilità delle colture sia tra i diversi campi che all’interno dello stesso campo. Questo approccio consente di ottimizzare la produzione agricola, ridurre gli impatti negativi sull’ambiente e garantire una fornitura continua di alimenti di alta qualità, contribuendo, così, a soddisfare le crescenti esigenze alimentari globali in modo sostenibile.

Affrontare questi problemi richiede un approccio integrato che coinvolga la ricerca scientifica, la formazione degli agricoltori, la promozione di pratiche agricole sostenibili e l’implementazione di regolamenti sanitari e ambientali rigorosi. Solo attraverso un impegno globale e coordinato è possibile affrontare efficacemente queste sfide e garantire una produzione agricola sicura, sostenibile e in grado di rispondere alle crescenti esigenze alimentari della popolazione mondiale.

L’agricoltura moderna affronta la sfida di mantenere l’equilibrio nell’ecosistema, evitando l’accumulo eccessivo di sostanze chimiche che possono danneggiare l’ambiente. La soluzione a questi problemi non può essere standardizzata poiché variano in base alle specifiche condizioni e alle manifestazioni locali. Per affrontare queste sfide complesse, è necessario adottare un approccio ecologico completo e flessibile, basato sull’osservazione e sull’analisi continua di tutti gli aspetti del sistema agricolo.

Un’opzione per mitigare questa situazione consiste nell’utilizzo dell’Intelligenza Artificiale, in particolare il Machine Learning. Quest’ultimo può fornire agli agricoltori dati e informazioni preziose per ottimizzare la produzione di colture, ridurre i costi iniziali e gestire le perdite dovute a calamità naturali. Questo approccio intelligente può contribuire a rendere l’agricoltura più efficiente ed ecologicamente sostenibile.

Un esempio di questo concetto è stato esplorato da Gomes e Leta nel 2012, che hanno studiato l’applicazione di tecniche informatiche nel settore agricolo e alimentare per migliorare la qualità dei prodotti. Allo stesso modo, Davies nel 2009 ha esaminato l’uso della visione artificiale e delle sue applicazioni nel settore agroalimentare. Queste ricerche dimostrano l’innovazione tecnologica, come quella apportata dall’Intelligenza Artificiale, possa portare a miglioramenti significativi all’agricoltura moderna.

1.3.2 Il Machine Learning

L’apprendimento automatico è un campo interdisciplinare che fonde l’informatica e la statistica, ponendosi come obiettivo l’analisi e la classificazione dei dati, svolgendo compiti che in genere richiederebbero l’intervento umano. Per raggiungere questo obiettivo, è necessario istruire i calcolatori in modo da risolvere problemi del mondo reale con la massima precisione possibile.

Di seguito, verranno illustrate le tre principali modalità di apprendimento automatico utilizzate negli algoritmi di Machine Learning, ovvero:

- Supervised Learning;
- Unsupervised Learning;
- Reinforcement Learning ;

Supervised Learning

L’apprendimento supervisionato, o *supervised learning*, come suggerisce il nome, richiede la guida di un supervisore per eseguire il compito. In questo tipo di apprendimento, una macchina viene addestrata utilizzando dati precedentemente raccolti e annotati, chiamati *dati etichettati*. Questi dati consentono all’algoritmo di apprendimento supervisionato di analizzare il set di addestramento e produrre risultati corretti basati sulle etichette fornite.

L’apprendimento supervisionato si suddivide in due categorie principali: la *regressione* e la *classificazione*. La regressione è una tecnica utilizzata per stabilire la relazione tra variabili indipendenti e variabili dipendenti, consentendo di predire valori numerici. D’altra parte, la classificazione consiste nel suddividere i dati in classi specifiche e distinte, assegnando un’etichetta a ciascuna classe.

L’approccio supervisionato è ampiamente utilizzato in applicazioni in cui è fondamentale effettuare previsioni o assegnare classificazioni basate su dati storici conosciuti e risultati noti, come nel riconoscimento delle immagini, nell’analisi del testo, nella diagnosi medica e in molte altre sfide decisionali.

Unsupervised Learning

L’apprendimento automatico non supervisionato, o *unsupervised learning*, è un tipo di approccio in cui il modello non richiede una guida esterna, ma estrae informazioni dai dati stessi, principalmente da dati non etichettati. Gli algoritmi di apprendimento non supervisionato sono spesso utilizzati in compiti complessi e possono generare risultati eccellenti, in particolare quando si tratta di scoprire modelli nascosti nei dati. Gli algoritmi di apprendimento

non supervisionato sono impiegati per scoprire pattern, cluster o associazioni all’interno dei dati in modo autonomo.

Questo tipo di apprendimento è stato applicato anche nell’agricoltura di precisione, dimostrando la sua utilità nell’analisi dei dati agricoli complessi. In generale, l’apprendimento non supervisionato può essere suddiviso in due categorie principali, ovvero il *clustering* (raggruppamento di dati simili) e l’*associazione* (individuazione di relazioni tra variabili o elementi nei dati).

Reinforcement Learning

Nell’apprendimento per rinforzo, o *reinforcement learning*, l’agente è dotato della capacità di interagire con l’ambiente e di migliorare le sue azioni nel tempo. Questo tipo di apprendimento si basa spesso su prove ed errori, in cui l’agente sperimenta diverse azioni per trovare una strategia che massimizzi il suo “rinforzo” o la sua ricompensa. L’apprendimento per rinforzo è particolarmente utile quando non esiste un metodo definito per eseguire un task. Infatti, l’agente deve seguire regole o strategie specifiche per svolgere efficacemente il suo task. In questo tipo di apprendimento, non sono richieste etichette o dati pre-etichettati.

Esistono due tipi principali di apprendimento per rinforzo: il *rinforzo positivo*, in cui l’agente è ricompensato per azioni corrette o desiderate, e il *rinforzo negativo*, in cui l’agente è punito o riceve un feedback negativo per azioni indesiderate. Questo feedback guida l’agente nell’apprendimento di strategie ottimali.

1.3.3 Algoritmi Machine Learning

L’apprendimento automatico, o *Machine Learning*, è un campo in continua crescita, con diversi autori che sfruttano una vasta gamma di algoritmi per risolvere problemi complessi. Questo campo è al centro di una ricerca attiva, con numerosi studi in corso. Nel seguito, daremo uno sguardo alle principali tecniche di Machine Learning.

Artificial Neural Network

Le *Artificial Neural Network* (ANN) sono una classe di modelli di apprendimento automatico ispirati al funzionamento del cervello umano. Queste reti sono composte da un gran numero di elementi chiamati neuroni artificiali, o nodi, che operano in modo simile ai neuroni biologici. Ogni neurone prende decisioni semplici basate sui dati di input e trasmette queste decisioni ad altri neuroni attraverso connessioni pesate. L’interconnessione di questi neuroni è conosciuta come *struttura* della rete neurale.

Una rete neurale può essere poco o molto profonda, a seconda del numero di strati di neuroni. Una rete neurale poco profonda ha solitamente tre strati principali:

- il *layer di input*;
- il *layer nascosto*;
- il *layer di output*.

Il layer di input riceve i dati in ingresso, il layer di output restituisce i risultati finali, ci possono essere uno o più layer nascosti tra di essi. Questi strati nascosti sono responsabili di elaborare e apprendere rappresentazioni complesse dai dati di input.

Support Vector Machine

Il *Support Vector Machine (SVM)* è un algoritmo ampiamente utilizzato nel campo dell’apprendimento automatico per risolvere problemi di classificazione e regressione. L’obiettivo principale dell’SVM è creare un confine decisionale efficace in uno spazio ad alta dimensione per separare le diverse classi dei dati. Questo confine decisionale è comunemente noto come *iperpiano*.

Clustering

Il *clustering* implica la suddivisione dei dati in gruppi in base alle loro somiglianze e differenze intrinseche. Fondamentalmente, si tratta di organizzare dati simili in insiemi distinti. Il clustering riveste un’importanza notevole, poiché facilita il raggruppamento dei dati precedentemente non strutturati.

I metodi di clustering includono approcci basati sulla *densità* e sulla *gerarchia*. Gli approcci basati sulla densità identificano i cluster come regioni dense all’interno dello spazio dati, con una notevole precisione e la capacità di unire cluster simili. Gli approcci basati sulla gerarchia creano una struttura ad albero basata sulla gerarchia dei cluster, suddividendoli in categorie.

Decision Tree

Il *decision tree*, o *albero decisionale*, è uno strumento di modellazione analitica ampiamente utilizzato in vari settori. Questi alberi vengono costruiti attraverso un approccio algoritmico che scomponete il set di dati in base a diverse condizioni. Nei modelli di alberi decisionali, la suddivisione dei dati si basa su criteri specifici e avviene in modo sequenziale. Questo algoritmo è composto da due componenti principali: i *nodi decisionali* e le *foglie*. I nodi decisionali sono i punti in cui avviene la suddivisione dei dati, mentre le foglie rappresentano i risultati finali.

Principle Component Analysis

La *Principle Component Analysis (PCA)* è una procedura statistica che trasforma le variabili correlate in variabili non correlate attraverso una trasformazione ortogonale. Viene utilizzata per esaminare le relazioni tra un insieme di variabili. Questo algoritmo è particolarmente utile quando si ha a che fare con un ampio insieme di variabili interconnesse e si desidera selezionare un sottoinsieme che sia il più informativo possibile per la creazione di un modello.

1.3.4 Applicazioni del machine learning in agricoltura

Alcune applicazioni del machine learning in agricoltura sono le seguenti:

Previsione delle rese

L’agricoltura coinvolge numerosi fattori che possono influenzare i risultati ottimali. La previsione del rendimento delle colture è uno di questi aspetti critici, che considera parametri come la fertilità del terreno, le tecniche di irrigazione, le condizioni meteorologiche e la gestione dei parassiti. La corretta gestione di questi quattro elementi è fondamentale per la protezione e il massimo rendimento del raccolto. Nell’ambito agricolo, sono stati adottati modelli di apprendimento automatico per migliorare la previsione e la gestione di tali fattori.

L’applicazione dell’apprendimento automatico ha dimostrato di essere estremamente utile nella gestione delle coltivazioni di caffè. Grazie a questa tecnologia, è possibile effettuare il conteggio dei semi di caffè su un ramo e suddividere i frutti di caffè in tre categorie: raccolti,

non raccolti e semi privi di maturazione. Inoltre, è possibile stimare con precisione il peso dei semi e calcolare la percentuale di maturazione dei semi di caffè.

Ramos ed altri nel 2017 hanno presentato un sistema di visione artificiale (MVS) che consente il conteggio automatico dei frutti di caffè direttamente dall’albero di caffè. Questo sistema si è dimostrato particolarmente affidabile, in quanto ha ottenuto una correlazione significativamente elevata, pari a 0,90, tra le stime di conteggio dei semi effettuate mediante MVS e la realtà.

Questa tecnologia non solo semplifica il processo di monitoraggio delle coltivazioni di caffè ma può anche contribuire a migliorare la gestione delle risorse e a ottimizzare la produzione agricola.

Rilevamento dei parassiti e malattie

Il controllo dei parassiti e delle malattie rappresenta una delle sfide principali nell’agricoltura contemporanea. Un approccio comunemente adottato è la diffusione uniforme di pesticidi sulle colture, sebbene ciò sia costoso ed implichi rischi ambientali, come la contaminazione delle risorse idriche e l’impatto sulla fauna selvatica e sull’ecosistema.

Il lavoro condotto da Ebrahimi nel 2017 ha portato allo sviluppo di una macchina in grado di identificare i parassiti all’interno di un ambiente di serra attraverso l’analisi delle immagini. Per questa identificazione, è stato utilizzato con successo il metodo SVM (Support Vector Machine) per la classificazione e il rilevamento dei parassiti.

L’approccio combinato di elaborazione delle immagini e l’impiego del metodo SVM, considerando in modo adeguato la provincia e l’indice di colore, si è rivelato altamente efficace nel rilevare i parassiti con un’elevata efficienza. Questo rappresenta un importante passo avanti nella gestione sostenibile delle coltivazioni in serra, consentendo il rilevamento precoce dei parassiti e, di conseguenza, un controllo più mirato ed efficiente.

Rilevamento delle erbacce

La prevenzione delle erbe infestanti è cruciale per garantire una buona resa nelle coltivazioni. Tuttavia, identificare e prevenire con precisione le erbe infestanti può essere una sfida, poiché spesso sono simili alle colture. L’apprendimento automatico basato su sensori rappresenta un approccio efficace che consente di rilevare ed evitare le erbe infestanti con precisione, riducendo i costi e minimizzando l’impatto sull’ambiente.

Nel lavoro condotto da Pantazi nel 2017, è stata impiegata la tecnologia di telerilevamento per distinguere tra diverse specie di erbe infestanti e per creare mappe operative di tali infestazioni. In particolare, è stata utilizzata una telecamera multivisionale ad alta risoluzione montata su un sistema aereo senza pilota (UAS) per esporre e mappare le aree infestate da *Silybum Marianum*, una specie di erba infestante.

L’uso di questa tecnologia avanzata ha permesso di ottenere immagini dettagliate e una mappatura precisa delle macchie di erbe infestanti, contribuendo, così, a una gestione più efficace e mirata delle infestazioni nelle coltivazioni.

Gestione del suolo

La gestione del suolo riveste un ruolo cruciale nell’ottimizzazione della resa delle colture, nella promozione della stabilità ecologica e nella salvaguardia della salute umana, sia in modo diretto che indiretto. Il suolo è una risorsa naturale estremamente diversificata, caratterizzata da processi complessi e intricati meccanismi. La temperatura del suolo stessa svolge un ruolo essenziale nell’analisi accurata delle variazioni climatiche di una determinata area e nel suo comportamento ecologico.

Gli algoritmi di apprendimento automatico hanno un ruolo di notevole importanza nella misurazione della temperatura e dell’umidità del suolo. Questi strumenti sono fondamentali per comprendere le dinamiche degli ecosistemi e, per valutare il loro impatto sull’agricoltura.

Nel lavoro condotto da Ghosh e Koley nel 2014, è stata introdotta una nuova tecnica chiamata *back propagation network* (rete a propagazione all’indietro) che ha dimostrato di ottenere risultati migliori nel processo di identificazione delle caratteristiche del suolo rispetto all’utilizzo del tradizionale modello di regressione multivariata. Il funzionamento di questa rete a propagazione all’indietro consiste nell’addestrare un modello specifico per riconoscere le proprietà del suolo desiderate in base a determinate colture.

Questo approccio, basato su reti neurali artificiali, offre un modo più efficace e preciso per analizzare le caratteristiche del suolo e può contribuire a migliorare la gestione e la produttività agricola.

Riconoscimento delle piante

L’apprendimento automatico offre un approccio più avanzato rispetto al metodo convenzionale di classificazione delle piante, che si basa sul confronto tra forma e colore delle foglie. L’uso dell’apprendimento automatico permette di ottenere risultati più precisi e rapidi analizzando la morfologia delle venature delle foglie; tale analisi fornisce informazioni dettagliate sulle caratteristiche delle foglie stesse. L’obiettivo principale di questo approccio è automatizzare il riconoscimento e la categorizzazione delle diverse varietà di piante, riducendo al minimo la necessità di intervento umano e accelerando il processo di categorizzazione.

Nel lavoro condotto da Grinblat nel 2016 è stata utilizzata una rete di convoluzione profonda per affrontare il problema dell’identificazione delle piante basata sui modelli delle vene delle foglie. Nello specifico, gli autori hanno studiato tre diverse specie di legumi: il fagiolo bianco, il fagiolo rosso e la soia, utilizzando la morfologia delle vene delle foglie come principale indicatore. Questo approccio si è dimostrato uno strumento significativo nell’identificazione delle piante, superando le limitazioni legate al riconoscimento basato solo sul colore e sulla forma delle foglie.

Gestione della qualità del raccolto

Per massimizzare il valore del raccolto e minimizzare gli sprechi, è essenziale classificare la qualità del raccolto con un margine di errore minimo. Questo processo implica lo sviluppo della penultima sotto-categoria della coltura al fine di identificare in modo accurato le caratteristiche associate a una specifica classe di coltura.

Nel lavoro condotto da Zhang nel 2017 è stato sviluppato un modello per rilevare e classificare il materiale estraneo, sia botanico che non botanico, che si radica nella lanugine di cotone durante il processo di raccolta.

Gestione dell’irrigazione

L’irrigazione riveste un ruolo essenziale nell’agricoltura, contribuendo in modo significativo alla produttività dei raccolti. Tuttavia, è fondamentale che essa sia gestita in modo equilibrato, evitando sia l’eccesso che la carenza di acqua. Per mantenere questo equilibrio, è importante prendere in considerazione diversi fattori, tra cui il tipo di suolo, la topografia del terreno, le condizioni meteorologiche, il tipo di coltura e la qualità dell’acqua disponibile.

Il *neuro drip*, sviluppato da Hinnell nel 2010, è un algoritmo basato su reti neurali artificiali (ANN) implementato in Excel. Questo strumento è stato progettato per analizzare e illustrare in modo rapido i modelli di bagnatura del suolo causati dai gocciolatori di superficie utilizzati nell’irrigazione agricola.

Benessere degli animali

Il settore del benessere animale è dedicato alla salute e al benessere degli animali, con l’obiettivo di preservare l’equilibrio degli ecosistemi. Una delle applicazioni chiave dell’apprendimento automatico in questo campo è il monitoraggio del comportamento degli animali nelle prime fasi di esposizione alle infezioni.

Dutta ha sviluppato un approccio di apprendimento automatico a due fasi che si è dimostrato efficace per la classificazione del comportamento dei bovini. Questo metodo si basa sull’uso di tecnologie di sensori per il bestiame e classificatori appositamente costruiti al fine di classificare e analizzare i cambiamenti comportamentali dei bovini, con l’obiettivo di migliorare la loro alimentazione.

Previsioni sul bestiame

L’applicazione dell’apprendimento automatico nella produzione di animali da allevamento è fondamentale per valutare in modo accurato i bilanci monetari. Questo approccio permette ai produttori di ottenere informazioni basate sul monitoraggio della linea di produzione, contribuendo così a massimizzare i profitti. Gli algoritmi di apprendimento automatico hanno dimostrato di essere in grado di rilevare e segnalare tempestivamente i problemi, offrendo informazioni preventive ai produttori.

Nello studio condotto da Morales nel 2016 è stato esaminato il riconoscimento della produzione di uova utilizzando un approccio basato su SVM. Questo metodo è stato sviluppato per identificare potenziali problemi nella produzione di uova utilizzando dati sulla produzione di uova da parte delle galline ovaiole. L’uso di una configurazione ottimale dei parametri in un modello SVM consente di emettere un avviso con un giorno di anticipo, il che può essere prezioso per diagnosticare precocemente i sintomi clinici e prevenire eventuali problemi nella produzione di uova.

1.3.5 Il Deep Learning

Il *Deep Learning* è una metodologia moderna che ha dimostrato successo in diverse aree, tra cui l’elaborazione delle immagini e la classificazione dei testi. La sua alta efficienza in vari settori ha portato alla sua applicazione anche nell’ambito agricolo. Il Deep Learning coinvolge l’utilizzo di reti neurali con numerosi strati, progettate per eseguire task complessi. Alcuni modelli di Deep Learning hanno ottenuto risultati eccezionali che superano le capacità umane su larga scala. Ogni strato di una rete Deep Learning utilizza l’output del precedente come input, e l’intera rete viene addestrata come un’unica catena.

Per sfruttarlo al meglio, esistono piattaforme apposite che aiutano gli utenti a costruire varie architetture di Deep Learning o ad applicarlo a diverse applicazioni aziendali, tramite app e servizi. Una delle principali differenze tra l’apprendimento automatico e il Deep Learning è la quantità di dati necessaria per la classificazione: il Deep Learning richiede tipicamente un maggior numero di dati rispetto all’apprendimento automatico, che può operare con dataset più limitati.

Alcuni dei principali strumenti di Deep Learning includono *Theano*, *Keras*, *TensorFlow*, *PyTorch* e diversi toolbox. Questi strumenti forniscono agli sviluppatori le risorse necessarie per implementare modelli di deep learning e applicarli con successo a una vasta gamma di problemi e applicazioni.

Convolution Neural Networks

Le *Convolution Neural Network* (CNN) sono una tipologia di reti neurali profonde che si basano su una struttura ispirata alla corteccia animale. Queste reti sono progettate per il

riconoscimento di pattern all’interno di immagini e video, nonché per l’elaborazione del linguaggio naturale e altre applicazioni. Una CNN utilizza una serie di strati, tra cui strati di convoluzione, per identificare ed estrarre caratteristiche rilevanti dalle immagini.

Nelle CNN vengono spesso utilizzate funzioni di attivazione, come la *Rectified Linear Unit (ReLU)*, per introdurre non linearità nel modello. La *convoluzione* è il processo chiave in una CNN e coinvolge l’applicazione di filtri apprendibili alle immagini di input, consentendo alla rete di rilevare caratteristiche rilevanti.

Per migliorare l’accuratezza delle CNN, può essere utilizzata una tecnica chiamata *aumento dei dati*, che consiste nel generare varianti dei dati di addestramento esistenti introducendo piccole modifiche o trasformazioni, come rotazioni o riflessi. Questo aiuta la rete a essere più robusta e ad avere una migliore capacità di generalizzazione.

Le CNN trovano applicazione in una vasta gamma di campi, tra cui il riconoscimento facciale, l’analisi di documenti, la raccolta e l’analisi di dati storici e ambientali, la previsione meteorologica, e persino, la pubblicità solo per citarne alcuni. Due sono particolarmente efficaci quando si tratta di compiti che coinvolgono dati visivi, come il riconoscimento di oggetti in immagini o video.

Le CNN sono ampiamente utilizzate nell’agricoltura grazie alla loro forte capacità di elaborazione delle immagini. Le principali applicazioni del Deep Learning nell’agricoltura possono essere catalogate come classificazione di piante o colture, previsione di parassiti e rese, raccolta con robot, monitoraggio di disastri, etc. In particolare, i modelli di riconoscimento delle malattie delle piante possono essere applicati tramite immagini delle foglie e classificazione dei pattern.

Il *Berkley Vision and Learning Centre* ha sviluppato un innovativo framework di Deep Learning per costruire un modello di rilevamento delle malattie delle piante. Questo sistema è in grado di identificare circa 10-15 casi di foglie malate rispetto alle foglie sane, ed è in grado anche di separare le foglie delle piante dall’ambiente circostante. Nel 2007, è stato sviluppato un approccio per il controllo e l’identificazione delle erbacce che è una combinazione di CNN e apprendimento delle caratteristiche K-means. I modelli manuali per il rilevamento delle erbacce portano ad un riconoscimento errato ed a una debole capacità di estrazione delle caratteristiche. Uno dei modelli CNN ampiamente utilizzati nella classificazione delle piante è *Alexnet*. Sulla base dei risultati sperimentali di Alexnet, possiamo dire che l’architettura CNN supera gli algoritmi di apprendimento automatico, ovvero le strutture realizzate manualmente, a causa dell’ingiustizia delle fasi fonologiche.

Per la divisione delle immagini ottiche e il successivo ripristino delle informazioni mancanti in una serie temporale di immagini satellitari, vengono utilizzate mappe Kohonen auto-organizzate. In questo metodo, per la configurazione del post-processing, vengono utilizzate l’analisi geo-spaziale e diverse tecniche di filtraggio. Anche se le CNN hanno diverse applicazioni, affrontano molte sfide che ne hanno rallentato l’applicazione nella classificazione delle piante. Ad esempio, ogni pixel delle immagini satellitari a bordo spaziale SAR è caratterizzato dalla fase di retro-diffusione e dall’intensità in molteplici polarizzazioni.

Per la previsione delle rese e la raccolta con robot, il conteggio dei frutti è uno dei fattori importanti. Non possiamo ottenere risultati soddisfacenti attraverso il conteggio tradizionale o il conteggio delle immagini video o fotografiche, e anche questi processi richiedono tempo. La pre-elaborazione di questo tipo di immagini è impegnativa a causa dell’occlusione e dell’illuminazione. Nel 2018, Hansen e il suo team hanno introdotto una tecnica per identificare gli animali da allevamento come i maiali utilizzando la funzione di riconoscimento facciale delle CNN. In passato, venivano utilizzati tag di identificazione a radiofrequenza per rilevare gli animali, il che era un lavoro complicato.

Per accompagnare una rete completamente convoluzionale, è stata proposta una tecnica nota come *rilevamento di blob*. Il primo passo è raccogliere etichette create dall’uomo da

un insieme di immagini raffiguranti della frutta, e quindi addestrare questo modello per la segmentazione delle immagini. Successivamente, la CNN è utilizzata per conteggiare le immagini biforcate e fornire un’approssimazione del numero di frutti. L’ultima fase del lavoro prevede l’applicazione di un’equazione di regressione per mappare la stima del conteggio intermedio dei frutti al conteggio finale delle etichette generate dall’uomo. La combinazione del Deep Learning con il rilevamento di blob aumenta l’accuratezza e l’efficienza.

Il metodo di classificazione del suolo è utilizzato per identificare l’uso e la copertura del suolo, per la valutazione dei rischi legati ai disastri e per l’agricoltura e il cibo. L’idea generale del metodo di deep learning è quella di integrare le informazioni sviluppate da diverse fonti eterogenee utilizzando tecniche di apprendimento automatico per fornire capacità di elaborazione delle informazioni e rappresentazione visiva. Questo processo include quattro fasi:

1. filtraggio del rumore e clustering dei dati;
2. pulizia della copertura terrestre;
3. post-elaborazione delle mappe;
4. analisi geo-spaziale.

Kussul ed altri hanno introdotto un approccio multi-livello di apprendimento profondo per la classificazione dell’uso del suolo e dei tipi di coltivazioni utilizzando immagini satellitari multi-temporali multi-sorgente.

Oggi, nel settore agricolo stanno emergendo nuove tecnologie; ad esempio, vengono utilizzati veicoli aerei senza pilota per l’elaborazione di immagini di alta qualità. Per gli agricoltori, gestire macchine altamente autonome è piuttosto difficile. Pertanto, la rilevazione automatica dei rischi in tempo reale con queste macchine, con un’alta affidabilità, diventa una necessità. Per un uso sostenibile del territorio, è necessario considerare alcune condizioni, come la pianificazione per la riduzione delle emissioni di CO₂, la diminuzione del degrado del suolo e il miglioramento dei rendimenti economici utilizzando dati preziosi dai satelliti. Per la presa di decisioni nell’agricoltura di precisione e nell’agroindustria, le CNN e l’algoritmo genetico sono diventati metodi convenienti con l’uso di immagini satellitari tradotte. La previsione meteorologica è uno dei principali fattori per gli agricoltori, che può essere sfruttato utilizzando le CNN. Allo stesso modo, la stima del rendimento delle colture è uno dei principali fattori per gli agricoltori, i consumatori e il governo, che deve essere effettuato prima del raccolto delle colture. Le CNN non vengono utilizzate solo per la stima delle colture o per scopi agricoli, ma anche per classificare il comportamento degli animali.

Recurrent Neural Network

Le *Recurrent Neural Network* (RNN) sono un tipo di rete neurale in cui l’output generato in un passaggio temporale precedente viene utilizzato come input nel passaggio successivo. Questo approccio trova applicazioni in una serie di campi, come il riconoscimento vocale, il riconoscimento della scrittura, e l’analisi di sequenze di dati.

In particolare, le reti neurali generative basate su RNN sono in grado di generare automaticamente codici di programmazione che soddisfano un obiettivo predefinito. Il processo di funzionamento di una RNN coinvolge la presentazione di un input al modello. Questo input viene elaborato attraverso uno strato di input, quindi inviato a uno strato nascosto che svolge la modellazione della sequenza e l’addestramento in avanti o all’indietro. Anche se possono essere presenti più strati nascosti, l’ultimo strato nascosto trasmette il risultato elaborato allo strato di output.

Un’evoluzione importante delle RNN è rappresentata dalle *RNN a memoria a lungo e breve termine* (o *Long Short-Term Memory* o *LSTM*), che sono ampiamente utilizzate. Queste sono particolarmente efficaci per il trattamento di sequenze di dati che richiedono la memorizzazione di informazioni a lungo termine o dettagli relativi agli eventi più recenti. Le applicazioni delle RNN includono la modellazione e la previsione del linguaggio, il riconoscimento vocale, la traduzione automatica e il riconoscimento e la traduzione di immagini.

Le LSTM, o celle, sono l’ultima evoluzione delle RNN e gestiscono in modo intelligente l’input dello stato precedente decidendo quali informazioni conservare e quali scartare. Integrando informazioni dal passato, dal presente e dall’input attuale, sono in grado di fare previsioni più precise.

La classificazione della copertura del suolo è un’area chiave nell’agricoltura, che coinvolge il riconoscimento del tipo e della qualità del terreno. In passato, molte applicazioni si basavano su osservazioni mono-temporali; questi metodi dipendevano da vari fattori, come il tempo atmosferico.

Per affrontare le sfide legate alle reti neurali ricorrenti (RNN) è stato introdotto un modello noto come *NARX*, che sta per processo di modello auto-regressivo non lineare con input esogeni. In questo metodo, i valori di previsione precedenti sono considerati come input, mentre i valori attuali e precedenti sono considerati come input esogeni. Il sistema valuta non solo gli input indipendenti ma anche la risposta precedente del sistema, il che rende il sistema più potente. Utilizzando il modello NARX, è stato sviluppato un altro modello, il *NARXNN*, per l’analisi delle serie temporali dell’indice di area fogliare (LAI). Kurumatani nel 2018, ha proposto una tecnica per prevedere il prezzo dei prodotti agricoli utilizzando le RNN.

Le RNN vengono anche utilizzate per prevedere il tempo atmosferico. Biswas e altri nel 2014 hanno progettato tre modelli per la previsione del tempo: il modello di rete neurale auto-regressiva non lineare con input esogeni (NARX NN), il modello di ragionamento basato su casi e il modello di ragionamento basato su casi segmentati. Palangpour ed altri nel 2016 hanno sviluppato un modello per identificare la posizione degli animali nella foresta. In questo modello, è stato utilizzato l’algoritmo di ottimizzazione dello sciame di particelle combinato con il modello RNN; i risultati ottenuti da questo modello presentano meno errori.

Generative adversarial Networks

Le *Generative Adversarial Networks* (*GAN*) rappresentano un’innovazione significativa nell’ambito dell’apprendimento automatico. Si tratta di un tipo di apprendimento non supervisionato che incorpora tecniche avanzate per identificare somiglianze o modelli nei dati generati dal sistema. Le GAN sono modelli intelligenti che risolvono problemi di generazione di dati attraverso l’interazione di due sottomodelli nel contesto dell’apprendimento supervisionato. Questi sistemi generativi possono essere addestrati per creare rappresentazioni visive, come illustrazioni o immagini.

Le GAN costituiscono un campo affascinante in rapida crescita, grazie al loro potenziale nell’ambito della generazione di dati realistici. Questi modelli trovano applicazioni in svariati domini, ad esempio nella trasformazione di immagini, come la conversione di scene estive in scene invernali o di immagini diurne in immagini notturne, al fine di generare immagini che appaiono autentiche e foto-realistiche agli occhi degli osservatori.

La GAN è considerata una delle reti neurali più utili in molti campi. Principalmente, la GAN viene utilizzata per trovare la perdita di dettagli nell’elaborazione delle immagini causata dalla riduzione di campionamento. Quando un’immagine viene compressa, alcune informazioni possono andare perse, o la qualità di quell’immagine si deteriora; quindi potremmo aver bisogno di recuperare tutti i dettagli originali. Per questo recupero, viene definita una *funzione di perdita perpetua* composta da perdita avversa e perdita di contenuto.

Questa funzione viene, quindi, confrontata con il *Mean Square Error (MSE)* pixel per pixel. Lavorando su un gran numero di immagini, questo modello è in grado di migliorare la qualità delle immagini altamente compresse. Ciò diventa importante in tutti i modelli che contengono lavori di elaborazione delle immagini, principalmente in agricoltura, perché alcune applicazioni dipendono dalle immagini da telerilevamento.

1.3.6 Vantaggi e svantaggi in agricoltura

In generale, nell’apprendimento automatico, non è facile analizzare i dati non strutturati. Per questo tipo di analisi di dati, l’applicazione dei metodi Deep Learning risulta più utile, in quanto consente di utilizzare diversi formati di dati per far funzionare gli algoritmi. In generale, i lavoratori possono stancarsi, diventare negligenti o trascurare i dettagli, ma nei modelli Deep Learning questo non accade. L’algoritmo eseguirà migliaia di cicli di lavoro senza errori, il tutto in un breve lasso di tempo. Inoltre, la qualità del lavoro non verrà compromessa a meno che i dati inseriti dall’utente non presentino problemi.

Nell’approccio di apprendimento tradizionale, l’identificazione delle caratteristiche deve essere accurata, mentre i modelli Deep Learning hanno la capacità di creare nuove caratteristiche autonomamente. Di solito, la risoluzione dei problemi nell’apprendimento automatico avviene suddividendo task complessi in task più piccoli e combinando i risultati di tutti i task minori per ottenere l’output finale. Invece, nel Deep Learning, i task vengono risolti su una base *end-to-end*, ovvero l’intero flusso di lavoro viene gestito in un’unica procedura.

Il Deep Learning richiede una grande quantità di dati o informazioni ed è costoso utilizzarne un modello. Uno dei principali svantaggi è che non siamo in grado di capire come avviene l’analisi all’interno del modello. Questo è spesso definito come *scatola nera*, ma in alcune circostanze è importante conoscere l’algoritmo di analisi, poiché l’interpretabilità è necessaria in alcuni settori.

Oggi, con l’espansione dell’apprendimento automatico in tutti i settori in modo notevole, la principale preoccupazione è che l’apprendimento automatico possa sostituire il lavoro umano e portare gli esseri umani verso la disoccupazione o la schiavitù.

1.3.7 IoT in agricoltura

Introduzione

L’*Internet delle cose*, o *Internet Of Things (IoT)*, trova applicazioni in una vasta gamma di settori; nel campo agricolo rappresenta un’opportunità per la modernizzazione e l’ottimizzazione delle pratiche agricole. Questa iniziativa ha aperto nuove prospettive nella ricerca avanzata nel settore. I dati sono raccolti da dispositivi mobili e sensori, e la trasmissione di tali dati avviene tramite tecnologie a radiofrequenza. Per garantire l’affidabilità e l’efficienza delle reti di sensori, vengono utilizzati metodi di verifica e ottimizzazione.

L’automazione dei processi e l’analisi predittiva sono possibili grazie all’impiego di approcci basati sui big data, il che permette di migliorare molte attività in tempo reale nel settore agricolo. Di seguito, si illustreranno le tecnologie IoT utilizzate per sviluppare sistemi di supporto per le diverse fasi delle operazioni agricole, esplorando le applicazioni dell’IoT in agricoltura, affrontando le sfide che questa tecnologia deve superare e delineando le opportunità future per il suo sviluppo nel settore agricolo.

Dispositivi IoT

Questa sezione esamina il ruolo dell’Internet delle cose (IoT) nell’ambito dell’agricoltura, esplora gli sforzi di ricerca correlati nonché l’applicazione delle tecnologie IoT in questo set-

tore. L’analisi critica svolge un ruolo guida nell’adozione di queste nuove idee sia nell’ambito dell’agricoltura urbana che in quello dell’agricoltura di precisione.

Il termine "Internet of Things" (IoT) è emerso come una tecnologia nel 1999 ed è da allora diventato un argomento di grande interesse in vari settori. Le previsioni indicano che il numero di oggetti connessi all’IoT crescerà notevolmente, passando dai 20 miliardi del 2015 a una stima di 75 miliardi entro il 2025. Questa evoluzione ha portato oggetti del mondo reale a essere connessi a Internet, consentendo il trasferimento di dati tramite il cloud.

Le capacità di comunicazione sono fondamentali per i dispositivi IoT, e ciò porta a numerose comunicazioni machine-to-machine (M2M). I dispositivi IoT possono scambiare dati tra di loro e con le applicazioni connesse, oltre a raccogliere dati da dispositivi e processi locali. Questi dati possono essere inviati a server centralizzati o in un cloud. I sistemi IoT offrono una vasta gamma di servizi, tra cui la modellazione, il controllo dei dispositivi e l’analisi dei dati. Il blocco di gestione dei dispositivi IoT è responsabile della gestione generale del sistema, mentre il blocco di sicurezza garantisce la protezione attraverso autenticazione e autorizzazione. Attualmente, il blocco delle applicazioni consente agli utenti di visualizzare e analizzare lo stato del sistema e di fare previsioni sul futuro.

Le applicazioni dell’IoT nell’ambito agricolo coprono una vasta gamma di scenari. I ricercatori hanno sviluppato applicazioni basate su reti di sensori scalari per il monitoraggio e il controllo delle infrastrutture agricole, come le serre, nonché per la cattura di immagini a distanza e il rilevamento di insetti utilizzando reti di sensori multimediali. Inoltre, sono state sviluppate applicazioni per il rilevamento delle malattie delle piante tramite l’elaborazione delle immagini, per la tracciabilità dei prodotti agricoli e per l’identificazione remota utilizzando reti basate su tag, come l’identificazione a radiofrequenza e la *Near Field Communication* (NFC).

Tuttavia, nei terreni agricoli, dove le colture e altri fattori possono variare la loro posizione nel tempo, possono verificarsi interferenze nella comunicazione tra i nodi dell’IoT. Per ovviare a questo problema, vengono utilizzati dispositivi *embedded* a bassa potenza, che sono noti per la loro stabilità a lungo termine. Il numero di dispositivi periferici, tra cui sensori e attuatori, da utilizzare dipende dal numero di ingressi/uscite digitali e analogiche richieste per il monitoraggio e il controllo delle attività agricole.

I dispositivi IoT vantano una capacità di autoconfigurazione che permette a un gran numero di essi di lavorare in modo collaborativo per fornire servizi specifici, come il monitoraggio meteorologico. Questi dispositivi sono in grado di configurarsi autonomamente, stabilire la connettività tra di loro e applicare gli aggiornamenti software più recenti con una minima interferenza da parte dell’utente. Inoltre, essi supportano una varietà di protocolli di comunicazione che consentono loro di interagire con altri dispositivi e sistemi.

Ciascun dispositivo IoT è identificato in modo univoco attraverso un identificatore specifico, come un *indirizzo IP* o un *Uniform Resource Identifier (URI)*. Grazie all’infrastruttura di controllo, configurazione e gestione, gli utenti possono accedere ai dispositivi IoT, verificare il loro stato e gestirli a distanza.

L’intelligenza decisionale integrata nei dispositivi IoT contribuisce all’efficienza energetica complessiva della rete, aumentandone, così, la durata operativa.

Benefici in agricoltura

L’Internet delle cose (IoT) svolge un ruolo cruciale nell’agricoltura. Gli agricoltori stanno adottando sia risorse hardware che software, sfruttando grandi quantità di dati nelle aree rurali, semi-rurali e urbane. L’uso di decisioni basate su dati in tempo reale è essenziale per ridurre i costi e ottimizzare l’utilizzo di risorse, e consente di misurare la qualità della produzione alimentare. Inoltre, l’IoT apre la possibilità di stabilire un rapporto diretto con i consumatori attraverso modelli di business adeguati.

L’adozione di monitoraggio delle colture basato sulla tecnologia contribuisce a ridurre i costi e le perdite di attrezzature agricole. I sensori vengono impiegati in sistemi di irrigazione automatica per monitorare fattori come umidità, temperatura e umidità del suolo, fornendo dati per ulteriori analisi e ottimizzazioni. I sistemi di supporto alle decisioni elaborano una vasta quantità di dati per migliorare la produttività e l’efficienza operativa, contribuendo così, a ridurre la necessità di manodopera intensiva. La ricerca suggerisce che l’agricoltura basata sulla tecnologia non solo aumenta la produttività, ma migliora anche le condizioni finanziarie e il benessere degli agricoltori. L’automazione completa dei sistemi di irrigazione aiuta a ottimizzare l’uso dell’acqua, grazie all’uso di reti di sensori wireless che raccolgono dati su umidità, temperatura e altre variabili da diverse zone del campo.

1.3.8 Applicazioni dell’IoT in agricoltura

La tecnologia svolge un ruolo cruciale nell’agricoltura. Questo sistema gestisce la raccolta dei dati, l’analisi statistica, l’archiviazione e il processo decisionale basato sui dati raccolti. Lo sviluppo delle tecnologie dei sensori e di altri dispositivi elettronici ha ridotto i costi e ha portato numerosi vantaggi alle pratiche agricole tradizionali. Questa evoluzione dai metodi agricoli convenzionali all’agricoltura di precisione e micro-precisione ha portato a notevoli miglioramenti nella produttività agricola.

L’uso dell’IoT in agricoltura si concentra sull’automazione e sulla fornitura di strumenti decisionali che integrano prodotti, conoscenze e servizi per migliorare la produttività, la qualità e i profitti agricoli.

L’IoT ha diverse applicazioni in agricoltura, ad esempio:

- l’uso corretto di pesticidi e fertilizzanti contribuisce ad aumentare la qualità dei raccolti e a ridurre i costi dell’agricoltura;
- l’uso dell’acqua in agricoltura e nelle attività connesse avviene attraverso sistemi di gestione dell’irrigazione ottimizzati;
- la tecnologia dei sensori wireless e le comunicazioni hanno contribuito al monitoraggio della qualità dell’acqua su parametri quali temperatura, pH, torbidità, conduttività e ossigeno dissolto;
- il monitoraggio della mandria di mucche e di altri animali al pascolo può essere effettuato utilizzando l’IoT;
- l’agricoltura e le serre sono collegate tra loro; l’aumento della temperatura del clima dovuto ai gas serra e il suo effetto sull’agricoltura sono monitorati;
- il monitoraggio del suolo viene effettuato utilizzando l’IoT;
- la conoscenza del suolo è importante per la produzione di cereali;
- la gestione della catena di approvvigionamento è necessaria per ottenere profitti che possono essere monitorati dall’IoT.

CAPITOLO 2

Il tracciamento dei prodotti

In questo capitolo si fornirà un'ampia visione della tracciabilità dei prodotti. Si inizierà descrivendo in dettaglio le responsabilità degli operatori del settore e le motivazioni che spingono al continuo sviluppo di questa disciplina. Successivamente, si metterà a confronto il tracciamento dei prodotti con altri settori economici ed, infine, si concluderà il paragrafo con uno sguardo al futuro.

In seguito, si esamineranno il tracciamento dei prodotti in diversi ambiti, come la sanità pubblica, l'industria, e la tracciabilità dei prodotti sfusi, inclusa la carne e il pollame.

Infine, l'ultima sezione del capitolo sarà dedicata alla blockchain, con un'analisi approfondita del suo utilizzo nella tracciabilità dei prodotti.

2.1 Introduzione

La **tracciabilità**, in termini generali, si riferisce alla capacità di seguire l'intera storia e i dettagli relativi al movimento dei prodotti. Questo concetto ha diverse applicazioni in vari contesti. La tracciabilità può essere impiegata per attestare la genealogia di individui o oggetti, mentre in alcuni settori è strettamente legata alla gestione della catena del freddo.

Tuttavia, nel contesto della tesi, la tracciabilità e la rintracciabilità dei prodotti si riferiscono specificamente al monitoraggio del percorso dei prodotti alimentari lungo l'intera catena di approvvigionamento. Questo processo comprende diverse fasi, tra cui la coltivazione, il raccolto, la produzione, l'imballaggio, lo stoccaggio, la distribuzione e la vendita ai consumatori finali. Le normative vigenti possono variare nei requisiti di registrazione in ciascuna di queste fasi, ma è importante notare che ogni punto della catena, sia esso un luogo di origine o destinazione del prodotto alimentare, contribuisce al tracciamento e alla rintracciabilità complessiva.

La rintracciabilità può essere focalizzata su due direzioni principali: la *trace forward*, che segue il flusso dei prodotti verso il consumatore finale, e la *trace back*, che risale il percorso dei prodotti fino alla loro origine. Poiché la rintracciabilità è strettamente connessa al movimento fisico dei prodotti, è spesso associata alle dinamiche delle catene di approvvigionamento e alla gestione logistica.

Tuttavia, quando si tratta di discutere di rintracciabilità nell'industria alimentare, il contesto predominante è quello della sicurezza alimentare. La comunità della salute pubblica e le autorità di regolamentazione pongono particolare enfasi sulla sicurezza alimentare, specialmente durante le indagini relative a focolai di malattie alimentari. In questi casi, la

rintracciabilità diventa uno strumento cruciale per individuare l'origine del problema e prevenirne la diffusione.

Al contempo, l'industria alimentare considera la sicurezza alimentare come una priorità fondamentale, si impegna principalmente a prevenire eventi di contaminazione, piuttosto che affidarsi esclusivamente alla rintracciabilità come soluzione retroattiva. In altre parole, l'obiettivo principale è evitare che eventi di sicurezza alimentare si verifichino durante le attività produttive.

2.1.1 Responsabilità per il tracciamento dei prodotti

La responsabilità della rintracciabilità degli alimenti all'interno di uno stabilimento di produzione alimentare può variare notevolmente. Può essere complesso individuare una singola entità all'interno dell'organizzazione che sia in grado di gestire tutte le fasi, compreso il ricevimento delle materie prime, la produzione, il dosaggio e la spedizione, e che, al contempo, abbia una piena comprensione dei problemi legati alla sicurezza alimentare che potrebbero richiedere l'attivazione del processo di tracciamento dei prodotti. Per questo motivo, sia l'architettura iniziale e l'istituzione di un sistema di tracciabilità, sia il processo di acquisizione delle informazioni necessarie, dovrebbero coinvolgere un team multidisciplinare.

I membri del team includono le seguenti aree:

- *Sicurezza alimentare*: la rintracciabilità diviene di fondamentale importanza in caso di eventi legati alla sicurezza alimentare che richiedono l'identificazione dell'origine dei materiali o, in fasi successive dell'indagine, la comprensione di dove siano stati venduti i materiali potenzialmente contaminati e come siano stati utilizzati. Tuttavia, data la natura stessa della rintracciabilità, il ruolo del professionista della sicurezza alimentare tende a essere periferico e consultivo rispetto al resto del team coinvolto. Il professionista della sicurezza alimentare deve essere competente nei requisiti normativi relativi alla documentazione e deve garantire che il sistema sia progettato con un livello di dettaglio e precisione che tuteli appieno la salute pubblica e, allo stesso tempo, consenta l'individuazione dei prodotti potenzialmente contaminati, minimizzando le perdite per la struttura coinvolta.
- *Catena di approvvigionamento e logistica*: la tracciabilità è centrata sulla gestione dei movimenti dei prodotti, di conseguenza, il professionista della catena di approvvigionamento svolge un ruolo cruciale nell'assicurare una tracciabilità efficace.
- *Tecnologia informatica*: la tracciabilità è strettamente connessa al movimento dei prodotti. Tuttavia, affinché esista un registro di rintracciabilità, è necessario registrare tale movimento. Sebbene nell'industria alimentare siano ancora diffusi i registri cartacei, c'è una forte preferenza verso l'adozione di registri elettronici. Molte aziende fanno uso di sistemi preesistenti, sviluppati su misura da professionisti per contenere i dati, inclusi quelli essenziali per garantire la rintracciabilità.
- *Contabilità*: poiché le merci vengono acquistate e vendute, alcune informazioni rilevanti per la tracciabilità possono essere incorporate nei sistemi contabili. In determinate circostanze, chi effettua l'acquisto o la vendita del prodotto potrebbe non coincidere con chi lo riceve o lo spedisce. Nonostante possa essere utile in certi casi seguire il denaro, è fondamentale tenere presente che la tracciabilità riguarda principalmente il monitoraggio del movimento di oggetti fisici.
- *Approvvigionamento e vendite*: coloro che acquistano e vendono prodotti alimentari e mangimi devono essere in grado di avere una chiara visione dell'identità dei propri

clienti e di determinare con precisione l'origine e la destinazione delle materie prime e dei prodotti finiti. Inoltre, se il prodotto presenta specifiche indicazioni che richiedono autenticazione, il personale coinvolto nell'approvvigionamento e nelle vendite manifesta un ulteriore interesse nella tracciabilità.

- *Spedizione e/o ricezione:* le persone responsabili di gestire i prodotti fisici svolgono un ruolo essenziale nell'assicurare una tracciabilità accurata. Le polizze di carico contengono informazioni cruciali riguardanti il luogo effettivo di spedizione e dovrebbero essere conservate, preferibilmente in formato elettronico, per garantire un registro affidabile.

2.1.2 Il tracciamento dei prodotti comparato agli altri settori

Spesso ci si domanda perché la tracciabilità degli alimenti può sembrare molto più complessa rispetto ad altri settori, dove ad esempio, le compagnie di trasporto merci gestiscono grandi volumi di spedizioni e i clienti possono monitorare la posizione precisa di un pacco online grazie a un codice di tracciamento. Anche i pezzi di ricambio delle automobili sono marcati con informazioni di tracciabilità. La differenza rispetto agli altri settori è dovuta a diversi aspetti che rendono la tracciabilità degli alimenti una sfida più complessa.

Tuttavia, è importante sottolineare che queste sfide non dovrebbero essere utilizzate come giustificazione per rinunciare agli sforzi volti a migliorare la tracciabilità nell'intero sistema alimentare. La disponibilità della tecnologia, combinata a eventuali cambiamenti normativi che potrebbero aumentare la visibilità della catena alimentare, potrebbero consentire un sistema di tracciabilità più veloce ed efficace per i prodotti alimentari.

Come in altri settori, anche gli alimenti viaggiano attraverso il mondo, talvolta attraversando diverse tappe prima di raggiungere il consumatore finale. La natura globale dell'approvvigionamento alimentare non rappresenta di per sé una sfida insormontabile per la tracciabilità. Tuttavia, ci sono diversi fattori che contribuiscono alla complessità della tracciabilità a livello globale.

Uno di questi fattori è la diversità delle normative relative alle informazioni che devono essere associate a un prodotto. Queste normative possono variare da paese a paese e spesso ci sono incoerenze nel modo in cui tali informazioni devono essere formattate e comunicate. Inoltre, la maturità tecnologica varia notevolmente in tutto il mondo, il che può complicare ulteriormente il processo di tracciamento dei prodotti su scala globale.

È importante notare che alcune economie emergenti hanno sfruttato le tecnologie più recenti per semplificare la tracciabilità, mentre altre nazioni, come gli Stati Uniti, che hanno sviluppato le proprie industrie durante la rivoluzione industriale, possono ancora avere impianti di produzione privi delle moderne capacità di telecomunicazione. La sfida, quindi, consiste nel creare un sistema di tracciabilità che possa essere applicato in modo coerente su scala globale, tenendo conto delle diverse normative e delle differenze tecnologiche tra le regioni.

L'approvvigionamento alimentare è un processo globale che coinvolge la distribuzione di alimenti e ingredienti attraverso lunghe distanze geografiche. Indipendentemente dalla distanza fisica percorsa dagli alimenti e dai loro componenti, un prodotto alimentare finito può essere composto da numerosi ingredienti, alcuni dei quali possono averne di propri. Ogni componente può essere stato tagliato, smistato, reimballato e manipolato da diversi membri della catena di approvvigionamento.

In questa intricata rete di produzione e distribuzione, ogni entità lungo il percorso deve acquisire informazioni sulla rintracciabilità e trasmetterle al membro successivo della catena di fornitura. Anche se la registrazione uno a uno sembra semplice quando ci sono solo pochi attori nella catena di fornitura, diventa incredibilmente complessa da gestire quando gli alimenti e gli ingredienti possono essere stati trasferiti e maneggiati decine di volte. La

tracciabilità efficace richiede un sistema robusto che sia in grado di gestire questa complessità e garantire che ogni passaggio lungo il percorso sia documentato e rintracciabile in modo accurato.

Il consumo di prodotti alimentari da parte di una famiglia tipica in un anno supera di gran lunga il numero di auto acquistate o dei pacchi consegnati a un determinato indirizzo nello stesso periodo. La produzione alimentare è un processo continuo e i prodotti alimentari hanno un ciclo di vita molto più breve rispetto ad altri beni di consumo.

Le famiglie acquistano generalmente poche auto in un anno, spesso una sola o nessuna, mentre le consegne a un indirizzo specifico possono variare notevolmente ma difficilmente raggiungono la quantità di prodotti alimentari consumati in un anno. La frequenza e l'entità dei consumi alimentari, insieme alla necessità di rifornire costantemente gli scaffali dei negozi, contribuiscono alla complessità e alla rapidità della catena di approvvigionamento alimentare. La tracciabilità degli alimenti è, quindi, fondamentale per garantire la qualità, la sicurezza e la conformità dei prodotti alimentari che consumiamo.

2.1.3 Le motivazioni allo sviluppo

Le applicazioni della rintracciabilità sono estremamente diverse. Quando ci si chiede il perché è importante la rintracciabilità dei prodotti alimentari ci si può aspettare una vasta gamma di risposte tra cui:

- la tracciabilità è necessaria per la sicurezza alimentare;
- la tracciabilità aiuta a migliorare l'efficienza operativa;
- la tracciabilità ci fornisce la visibilità necessaria per comunicare le informazioni richieste dai clienti;
- la tracciabilità aiuta ad autenticare le richieste di risarcimento.

2.1.4 Sviluppi futuri

Ci sono diversi fattori che convergono e che aumentano la probabilità che la tracciabilità degli alimenti continui a migliorare. In primo luogo, i recenti progressi tecnologici e infrastrutturali stanno contribuendo a rendere la tracciabilità degli alimenti più accessibile ed efficace. Sebbene l'adozione e l'implementazione possano essere ancora onerose per alcune aziende e in alcune regioni del mondo, è probabile che nel futuro questi ostacoli tecnologici vengano superati, come spesso avviene con altre applicazioni della tecnologia.

In secondo luogo, c'è un crescente interesse da parte dei consumatori nel conoscere ulteriori informazioni sui prodotti alimentari che acquistano, come la loro origine, gli allergeni, gli ingredienti e altri dettagli.

La crescente consapevolezza della scarsità delle risorse e la necessità di sfamare una popolazione in continua crescita (si prevede che raggiunga quasi dieci miliardi di persone entro il 2050) stanno spingendo l'industria alimentare verso una maggiore efficienza. La tracciabilità offre la visibilità necessaria per identificare le opportunità di miglioramento dell'efficienza in tutto il processo di produzione e distribuzione alimentare, contribuendo a rendere l'industria alimentare più sostenibile e in grado di soddisfare la crescente domanda alimentare globale.

2.2 Campi di utilizzo

2.2.1 Salute pubblica

Il concetto di rintracciabilità degli alimenti in risposta a focolai di malattie di origine alimentare o ad eventi di contaminazione alimentare ha una lunga storia. Già alla fine del 1800 e all'inizio del 1900, gli scienziati avevano stabilito dei collegamenti tra focolai di tubercolosi e streptococco e il consumo di latte crudo. Allo stesso modo, nei primi anni del 1900, i focolai di febbre tifoide sono stati associati al consumo di ostriche.

La rintracciabilità degli alimenti è un elemento fondamentale nella prevenzione e nella gestione di focolai di malattie di origine alimentare e nella garanzia della sicurezza degli alimenti per i consumatori.

All'inizio degli anni '80, numerosi focolai di Norovirus e di epatite A sono stati collegati al consumo di vongole veraci crude o poco cotte. L'inadeguatezza della marcatura dei molluschi ha complicato e rallentato le indagini e la risposta a questi focolai. Più tardi, negli anni '80 e '90, le uova in guscio sono state collegate a numerosi focolai di salmonella. Le indagini su questi focolai sono state difficili perché le informazioni richieste sui contenitori delle uova in guscio indicano dove le uova sono state confezionate, ma non necessariamente dove sono state deposte. A partire dagli anni '90 si sono verificati numerosi focolai di malattie di origine alimentare legati al consumo di prodotti freschi. Le indagini sulla tracciabilità dei prodotti freschi sono state molto difficili da realizzare a causa della complessità del sistema di produzione e distribuzione dei prodotti freschi e dell'inadeguatezza dei registri del sistema. I prodotti freschi di solito non sono accompagnati da informazioni sulla fonte. E quando ciò avviene, le informazioni vengono generalmente scartate dall'utilizzatore finale. Nel 2007, la contaminazione di cibo per animali e latte per l'infanzia con la melamina, motivata da ragioni economiche, ha portato a un'indagine internazionale. La ricerca è stata complicata da un complesso sistema di distribuzione e produzione e da una registrazione inadeguata.

Questi esempi forniscono un contesto cruciale per comprendere perché sono stati investiti tanto tempo ed impegno nell'ottimizzare la rintracciabilità degli alimenti dall'origine fino al punto di consumo. La ragione principale dietro questi sforzi risiede nella stretta relazione tra rintracciabilità e tutela della salute pubblica.

Quando le indagini sui focolai coinvolgono un alimento come veicolo di malattia, gli scienziati devono stabilire l'identità di tale alimento e prevenire ulteriori esposizioni a esso. È anche essenziale determinare il luogo e le circostanze in cui l'alimento è stato contaminato. Nella maggior parte dei casi, i focolai di malattie di origine alimentare segnalati negli Stati Uniti sono il risultato di errori commessi durante la preparazione del cibo, spesso in ristoranti. Tuttavia, se l'indagine rivela che la contaminazione non ha avuto luogo nel punto di preparazione, si avvia una procedura di rintracciabilità per individuare l'origine dell'alimento contaminato. Allo stesso modo, se un campione di prodotto alimentare risulta positivo per la presenza di agenti patogeni anche senza la conferma di casi di malattia, solitamente si avvia un'indagine e una rintracciabilità per prevenire potenziali focolai futuri.

Ogni indagine su un focolaio di malattia alimentare pone gli scienziati di fronte a una sfida delicata, spesso riassunta come il dilemma tra "correttezza e velocità". Questo conflitto è innescato dalla necessità di bilanciare l'urgenza legale ed etica di identificare rapidamente l'alimento contaminato e rimuoverlo dalla circolazione per prevenire ulteriori malattie, con la preoccupazione legale ed etica di non agire troppo rapidamente, identificando l'alimento errato e causando danni economici ingiustificati all'azienda e ai suoi dipendenti.

L'avvertire il pubblico di evitare un alimento che alla fine si scopre non essere stato contaminato non garantisce la protezione, poiché i consumatori potrebbero comunque consumare l'alimento effettivamente contaminato. Un errore del genere può anche minare la credibilità delle future indagini sui focolai presso il pubblico e l'industria. La pressione per

agire rapidamente può spingere le agenzie a basarsi su informazioni meno definitive, agendo con una certezza limitata nell'interesse della protezione della salute pubblica. D'altra parte, la necessità di essere corretti può portare a una riluttanza ad agire finché non si dispone di prove più solide, esponendo, così, il pubblico a un maggiore rischio di consumare alimenti contaminati e di sviluppare malattie.

La gestione efficace di questi dilemmi richiede una collaborazione attenta tra le agenzie di salute pubblica, l'industria alimentare e altre parti interessate per garantire una risposta rapida ed efficace ai focolai di malattia alimentare, senza compromettere la precisione delle indagini e la fiducia del pubblico.

Perché si conducono i tracciamenti

Le indagini di rintracciabilità rappresentano un tentativo mirato di ricostruire la catena di approvvigionamento alimentare di uno o più alimenti sospettati di essere all'origine di un focolaio di malattia. Quando più persone si ammalano in diverse località nello stesso periodo di tempo e il cibo è identificato come la fonte sospetta, viene avviata un'indagine di rintracciabilità. L'obiettivo principale di questa indagine è individuare un punto di "convergenza" nella catena di approvvigionamento, al fine di identificare la possibile fonte comune di malattia. In altre parole, gli investigatori cercano di risalire al punto in cui tutti i percorsi si incontrano, suggerendo così il possibile luogo o l'origine dell'incidente.

Questo processo di rintracciabilità può essere estremamente complesso, specialmente quando coinvolge una vasta gamma di prodotti alimentari e molteplici fornitori. Tuttavia, è fondamentale per identificare la fonte del problema, isolare i prodotti contaminati e prevenire ulteriori malattie, contribuendo, così, alla protezione della salute pubblica.

Uno degli obiettivi principali di un'indagine di rintracciabilità è quello di prevenire il verificarsi di ulteriori casi di malattia. A tal fine, è necessario che la rintracciabilità sia condotta nel modo più rapido e accurato possibile per individuare una potenziale fonte di contaminazione nella catena di approvvigionamento. La probabilità di interrompere la distribuzione di un prodotto contaminato è minore quando si tratta di prodotti a breve conservazione o ad alta rotazione, come i prodotti freschi, a meno che la fonte di contaminazione non sia continua (acqua di irrigazione, etc.), mentre la probabilità di interrompere la distribuzione di un prodotto a lunga conservazione, come il burro di arachidi, è maggiore.

Identificando la fonte comune dei prodotti contaminati nella catena di approvvigionamento, l'industria e le autorità di regolamentazione possono implementare misure efficaci per prevenire l'ingresso di ulteriori prodotti contaminati sul mercato e rimuovere quelli già contaminati attraverso un richiamo degli alimenti.

All'interno della comunità dei regolatori e della sanità pubblica, si è sviluppata una distinzione sulla base del "tipo" di indagine di rintracciamento che può essere condotta da un'agenzia regolatoria. Quando l'indagine epidemiologica non riesce a identificare in modo definitivo un singolo alimento sospetto associato ai casi di malattia in un focolaio, può essere condotta una rintracciabilità *investigativa* o *epidemiologica* su diversi alimenti per determinare se uno di essi mostra una convergenza nella catena di approvvigionamento che può spiegare la maggior parte delle malattie associate al focolaio. Questo tipo di rintracciamento è condotto con un'attenzione particolare alla rapidità e i dati possono essere raccolti via e-mail o via telefono nel tentativo di ricostruire rapidamente la catena di approvvigionamento e individuare un punto di convergenza.

A differenza di un rintracciamento epidemiologico, un rintracciamento *normativo* viene generalmente avviato quando esiste un forte indicatore che un singolo alimento è associato alle malattie in un focolaio. Nel caso di un rintracciamento normativo, il livello di raccolta delle prove è solitamente più completo: gli investigatori statali negli USA, ad esempio, visitano di persona ogni struttura che ha gestito il prodotto sospetto. Questo tipo di indagine mira

a esaminare attentamente la struttura e determinare se ci siano altre fonti di contaminazione plausibili che possano spiegare i casi di malattia nel focolaio, oltre all'alimento sospetto. Ad esempio, se l'alimento sospetto è stato conservato in un magazzino prima di essere distribuito ai negozi di alimentari della zona, gli investigatori cercheranno di determinare se ci sia stata una fonte di contaminazione all'interno del magazzino che potrebbe aver contaminato l'alimento, diventando così la fonte plausibile del focolaio.

Descrizione di un tracciamento in ambito sanitario

Un *tracciamento*, nel senso più puro del termine, è semplicemente un'estensione dell'indagine epidemiologica avviata dagli epidemiologi locali o statali e rappresenta uno sforzo per caratterizzare ulteriormente e in modo più accurato l'esposizione a un alimento sospetto per ogni caso di malattia in un focolaio. Di solito, un'indagine di rintracciamento inizia dopo che gli epidemiologi hanno analizzato i casi di malattia in un focolaio e hanno limitato il numero di alimenti sospetti.

A questo punto, gli investigatori delle agenzie statali responsabili della regolamentazione degli alimenti collaborano attivamente con gli epidemiologi per determinare con maggiore precisione i luoghi di esposizione più probabili associati a ciascun caso di focolaio. Questo processo implica anche la raccolta di dati relativi alla distribuzione dei prodotti alimentari nei suddetti luoghi. Ad esempio, una persona potrebbe aver acquistato un alimento contaminato presso un negozio di alimentari, mentre un'altra potrebbe aver consumato lo stesso alimento contaminato in un ristorante situato in una città o stato diverso. Durante un'indagine, ciascuno di questi luoghi in cui si è verificata l'esposizione al cibo sospetto costituisce una "tappa" nell'indagine di tracciabilità, rappresentando un punto chiave in cui uno o più casi sono stati esposti a tale alimento.

Nel corso delle indagini iniziali in una tappa, le autorità di regolamentazione raccolgono le fatture e le informazioni relative agli ordini di acquisto da ciascuna di queste sedi e iniziano il processo di tracciamento dei prodotti alimentari sospetti attraverso la catena di approvvigionamento. Nella maggior parte dei casi, gli alimenti consegnati e venduti nei punti vendita al dettaglio, come negozi di alimentari o ristoranti, possono essere ricondotti a un'azienda di distribuzione alimentare con un magazzino che può trovarsi nelle vicinanze o anche in stati diversi.

La maggior parte dei produttori e dei distributori di alimenti mantiene una documentazione completa delle transazioni alimentari con altre entità nella catena di approvvigionamento, compresi fornitori e clienti. Queste registrazioni sono state create e rese disponibili alle autorità di regolamentazione coinvolte nelle attività di rintracciamento principalmente perché documentano le transazioni e agevolano la gestione dei pagamenti tra le aziende della filiera alimentare. Gli investigatori di solito si riferiscono a questi documenti, che includono fatture, ordini di acquisto e polizze di carico, per mantenere la tracciabilità *esterna* all'interno della fornitura.

Gli investigatori possono utilizzare questi documenti esterni per tracciare l'alimento sospetto dal magazzino del distributore fino all'installazione di produzione in cui l'alimento è stato creato, considerando il gran numero di ingredienti utilizzati.

Le indagini di rintracciabilità spesso incontrano delle sfide quando si cerca di stabilire la tracciabilità *interna* di un distributore o di un produttore di alimenti. Fino a poco tempo fa, molte aziende operanti nella distribuzione e produzione alimentare non avevano processi o sistemi di gestione dei dati adeguati per mantenere una tracciabilità interna nella catena di approvvigionamento. La tracciabilità interna richiede che i trasformatori o i distributori di alimenti tengano traccia degli elementi interni che possono modificare l'identità o la configurazione del prodotto che vendono. Nel caso della produzione alimentare, la tracciabilità interna può implicare la registrazione e l'archiviazione di tutte le informazioni relative al

codice di lotto o alla partita degli ingredienti utilizzati (come cereali, sciroppo di mais, aromi, vitamine, etc.).

Se la tracciabilità interna non viene gestita con precisione e coerenza all'interno di un'azienda o per un determinato alimento oggetto di un'indagine, si verifica una perdita della completa tracciabilità nella catena di approvvigionamento. Questa mancanza di tracciabilità interna è particolarmente comune quando si tratta di alimenti che sono composti da numerosi ingredienti, prodotti sfusi e articoli freschi non confezionati.

Per proseguire con un'indagine di rintracciabilità in queste situazioni, gli investigatori devono stimare la probabilità che un prodotto sospetto sia stato spedito da o ricevuto in una determinata struttura, considerando un intervallo temporale che tenga conto del flusso del prodotto all'interno di quella struttura. Ad esempio, se un prodotto fresco viene generalmente conservato per due giorni in un magazzino refrigerato, l'investigatore potrebbe richiedere la documentazione riguardante le operazioni di ricevimento e spedizione del prodotto per una settimana a partire dalla data di interesse relativa al prodotto alimentare sospetto oggetto di rintracciabilità. Questo intervallo temporale dovrebbe includere tutte le transazioni sospette, sia quelle di ricevimento nel magazzino che quelle di spedizione dal magazzino, al fine di individuare le possibili fonti sospette del prodotto contaminato.

Le sfide del tracciamento dei prodotti in ambito sanitario

Se utilizzate come strumento per confermare un'ipotesi di esposizione epidemiologica, le indagini di rintracciamento dovrebbero essere condotte per la maggior parte dei focolai che si sospetta essere collegati ad un alimento distribuito commercialmente. L'applicazione più ampia delle indagini di rintracciamento per identificare le fonti dei focolai dovrebbe portare ad una maggiore scoperta delle fonti rispetto all'uso più limitato di queste indagini. Tuttavia, è importante notare che in entrambi i casi, per ogni indagine di rintracciamento avviata con successo, ve ne sono almeno altrettante che non riescono a individuare una convergenza nella catena di approvvigionamento o a identificare la fonte di un focolaio. Ci sono numerosi motivi per cui un'indagine di rintracciabilità può risultare complessa per un investigatore, sia dal punto di vista normativo che epidemiologico.

La scarsa qualità della documentazione rappresenta spesso un ostacolo rilevante nelle indagini di rintracciabilità che non raggiungono il successo desiderato. Molte aziende alimentari non mantengono i registri necessari per la rintracciabilità in modo adeguato. Questi registri dovrebbero consentire di documentare la provenienza del prodotto e da dove viene spedito in uscita.

Oltre alla mancanza di documenti o alla loro illeggibilità, i prodotti, specialmente quelli sfusi, possono essere identificati con nomi diversi da parte di ciascuna azienda coinvolta nella catena di fornitura. Questa variabilità nella nomenclatura per lo stesso prodotto può generare confusione e ritardi per gli investigatori che cercano di determinare se stanno ancora seguendo la pista del medesimo prodotto.

Anche i prodotti a breve durata di conservazione, come i prodotti freschi e i frutti di mare, possono presentare sfide nella rintracciabilità all'interno della catena di approvvigionamento. Questi articoli spesso non sono confezionati e vengono venduti sfusi o in casse con informazioni limitate sul codice del lotto associato al prodotto alimentare. Inoltre, quando le informazioni sul codice del lotto sono disponibili, la maggior parte dei distributori alimentari non ha la capacità di tracciare dettagliatamente queste informazioni a livello delle singole casse nei propri magazzini. La gestione dei prodotti si basa principalmente sul principio *first-in-first-out*, il che rende difficile agli investigatori determinare i punti vendita in cui il prodotto è stato spedito dal magazzino di distribuzione.

In aggiunta, i prodotti con breve durata di conservazione possono rapidamente esaurirsi dal mercato e talvolta essere completamente consumati dal pubblico prima che un'indagine di rintracciabilità possa essere avviata.

Anche i prodotti alimentari confezionati con una lunga durata di conservazione rappresentano una sfida per gli investigatori poiché le aziende sono tenute a conservare i registri solo per un periodo di 3-4 mesi dopo il completamento di una transazione. Questo diventa problematico quando si tratta di prodotti con una durata di conservazione di 12 mesi, in quanto le informazioni relative alle date di spedizione di tali prodotti possono sfuggire agli investigatori a causa della mancanza di registrazioni disponibili. Sebbene in generale sia più semplice tracciare i prodotti alimentari confezionati rispetto a quelli non confezionati, quando un focolaio è causato da un ingrediente contaminato presente in un prodotto confezionato, come ad esempio la pasta di arachidi utilizzata in un alimento complesso, diversi alimenti confezionati possono essere colpiti. Questo può rendere difficile individuare un punto di convergenza comune nella catena di approvvigionamento, a meno che l'investigatore non riesca a risalire all'origine dell'ingrediente contaminato.

La maggior parte delle aziende alimentari è familiare con il concetto di *richiamo* di un prodotto alimentare, ma spesso c'è una confusione nell'associare questa pratica con un'indagine di rintracciabilità. Nel caso di un richiamo di prodotto, un'azienda semplicemente identifica tutti i luoghi in cui ha distribuito il prodotto interessato, e queste informazioni sono generalmente facilmente reperibili nei propri sistemi di gestione dei dati. In alcuni casi, un'azienda potrebbe essere prudente e richiamare più prodotti di quelli effettivamente potenzialmente contaminati.

Tuttavia, l'indagine di rintracciabilità è un processo molto più specifico e dettagliato rispetto a un semplice richiamo di prodotto. In un'indagine di rintracciabilità, gli investigatori si concentrano su un numero limitato, spesso uno solo, di spedizioni sospette all'interno di un impianto. La specificità richiesta in un'indagine di rintracciabilità supera di gran lunga quella necessaria per un richiamo, e la maggior parte delle aziende incontra difficoltà nel riesaminare in dettaglio tutti i registri pertinenti e nell'identificare un numero limitato di spedizioni sospette all'interno della catena di approvvigionamento.

2.2.2 Industria

Il valore principale dei dati standardizzati risiede nella loro capacità di consentire ai partner commerciali di condividere informazioni e ottenere una visione chiara di ciò che accade lungo la catena di approvvigionamento. La tracciabilità rappresenta un'applicazione fondamentale di questa visibilità della catena di approvvigionamento, poiché utilizza dati basati su eventi per facilitare la rintracciabilità e il ritiro dei prodotti. Tuttavia, è importante notare che la tracciabilità rappresenta solo una delle molteplici applicazioni possibili per queste informazioni.

Le informazioni sulla visibilità della catena di approvvigionamento possono essere sfruttate per migliorare una vasta gamma di processi aziendali. Diverse applicazioni commerciali sfruttano i benefici di una maggiore visibilità della catena di approvvigionamento.

Le implementazioni della visibilità della catena di approvvigionamento forniscono diversi vantaggi a vari settori, come beni di consumo confezionati, vendita al dettaglio/alimentari, servizi di ristorazione, sanità, etc. L'industria alimentare in generale, e la categoria dei prodotti freschi in particolare, hanno un forte interesse a migliorare la visibilità della catena di approvvigionamento a causa di fattori unici del settore, come la natura deperibile dei prodotti, la volatilità dei mercati e la sensibilità alle sfide logistiche impreviste, come condizioni meteorologiche avverse e intoppi nei trasporti. L'implementazione della visibilità della catena di approvvigionamento può avanzare la sicurezza alimentare, rafforzare gli sforzi di

sostenibilità e migliorare l'efficienza dei processi aziendali, offrendo all'industria alimentare numerosi vantaggi, tra cui:

- efficienza operativa derivante dalla visibilità della catena di fornitura;
- miglioramenti sostanziali nei processi aziendali, come la riduzione del lavoro manuale grazie all'ampia adozione del *self-checkout*;
- una gestione più precisa degli stock, ordini più accurati, maggiore disponibilità dei prodotti e una migliore gestione delle discrepanze nell'inventario;
- riduzione dell'impatto economico delle crisi legate alla sicurezza alimentare mediante un rapido isolamento dei prodotti coinvolti e il ripristino della fiducia dei consumatori.

Si stima che negli USA questi vantaggi potrebbero equivalere a circa 3 miliardi di dollari soltanto per il settore alimentare fresco, basandosi su un'analisi prudente dei risultati, dei progetti pilota e dei casi di studio settoriali.

Benefici alla catena di approvvigionamento

I vantaggi della visibilità della catena di approvvigionamento non si limitano a specifici segmenti di essa. Nel 2013, l'*IFT (Institute of Food Technologies)* ha condotto un progetto pilota in collaborazione con l'industria alimentare per esaminare metodi di tracciabilità alimentare rapidi ed efficienti. Inoltre, ha preparato un dettagliato rapporto chiamato "*Rapporto pilota IFT*" per la *Food and Drug Administration (FDA)* degli Stati Uniti, in conformità con il *Food Safety Modernization Act (FSMA)*. Durante tali progetti pilota sono state condotte interviste ai partecipanti per comprendere i benefici ottenuti dai vari segmenti della catena di approvvigionamento. Oltre ai vantaggi della tracciabilità, i partecipanti ai progetti pilota hanno riportato numerosi altri benefici derivanti dalla visibilità della catena di approvvigionamento, tra cui:

- *Distributori*: maggiore capacità di gestione dell'inventario, che consente di fornire informazioni migliori alle forze di vendita, di migliorare i tassi di prelievo e di ridurre le differenze inventariali. Uno dei partecipanti al progetto pilota ha stimato un risparmio combinato sui costi e un aumento delle vendite di 500.000-600.000 dollari.
- *Spedizionieri*: visibilità in tempo reale della catena di approvvigionamento sui prodotti effettivamente confezionati sul campo, che ha contribuito a ridurre le situazioni di sovra-vendita o sottovendita giornaliera. Inoltre, la tracciabilità in tempo reale dal campo ai refrigeratori ha migliorato la capacità di assegnare priorità ai carichi destinati ai refrigeratori in base al momento della raccolta dei prodotti.
- *Rivenditori*: gestione dell'inventario più efficiente, con una maggiore precisione nelle scorte, un processo di selezione più efficiente e una migliore visibilità della catena di approvvigionamento.
- *Produttori*: aumento della produttività, maggiore precisione nell'assicurare che i prodotti corretti siano consegnati ai clienti appropriati e miglior efficienza nelle attività di fatturazione di back-office grazie alla capacità di identificare in modo più agevole i destinatari dei prodotti.

Benefici per la gestione dell'inventario

Oltre alla tracciabilità, le aziende che implementano la visibilità della catena di fornitura possono sfruttare i dati per migliorare l'efficienza operativa e ottimizzare la gestione dell'inventario, delle risorse e della qualità. Questi miglioramenti nella catena di fornitura possono portare a benefici concreti in termini di profitti. Ad esempio, l'ottimizzazione dei processi di gestione dell'inventario è di vitale importanza per i prodotti deperibili, dove la freschezza e la tempestività delle consegne sono fondamentali per mantenere la massima qualità dei prodotti alimentari.

Inoltre, l'implementazione di processi automatizzati per la visibilità della catena di fornitura può accelerare il movimento di qualsiasi tipo di prodotto attraverso la catena di fornitura, aumentando l'efficienza.

La visibilità della catena di fornitura apporta significativi miglioramenti ai processi di gestione dell'inventario e delle categorie, anche dal punto di vista della domanda. Per quanto riguarda la gestione dell'inventario, l'uso di numeri di lotto e date di ricevimento per identificare in modo univoco i prodotti facilita l'adozione di un approccio *first-in, first-out* per la gestione delle scorte. Questa filosofia può essere sfruttata per impostare avvisi relativi alla rotazione delle scorte, ottimizzando così la qualità e minimizzando il deterioramento dei prodotti. Inoltre, consente ai rivenditori di automatizzare in modo più efficiente le riduzioni dei prezzi man mano che le date di scadenza si avvicinano, evitando la vendita di prodotti scaduti. L'identificazione univoca dei prodotti e i dati aggiuntivi consentono anche di programmare automaticamente ribassi dei prezzi e di inviare avvisi sulle date di scadenza, inclusi avvisi relativi a prodotti richiamati.

Benefici per la brand reputation

La visibilità della catena di fornitura può contribuire a consolidare la reputazione del marchio e a instillare fiducia nei consumatori. Le informazioni accessibili sulla visibilità possono influenzare le decisioni che hanno un impatto sulla reputazione del marchio, migliorando, quindi, il processo decisionale. Ad esempio, una catena di ristoranti di portata nazionale ha avviato un ambizioso progetto finalizzato a garantire la tracciabilità completa della sua catena di fornitura, al fine di mantenere la promessa del marchio riguardo all'uso degli ingredienti migliori, all'approvvigionamento locale e alle pratiche commerciali sostenibili. L'azienda ha collaborato efficacemente con una vasta rete di fornitori partner per stabilire un sistema di tracciabilità aziendale, consentendo la condivisione di informazioni standardizzate sui prodotti in ogni fase della catena di fornitura.

Riduzione degli impatti finanziari di un richiamo

Un efficace sistema di visibilità nella catena di fornitura, associato a un robusto programma di tracciabilità, offre alle aziende l'opportunità di limitare l'impatto economico delle emergenze legate alla sicurezza alimentare, riducendo, così, gli sforzi necessari per ritirare i prodotti dal mercato. I richiami rappresentano un onere significativo per l'industria alimentare fresca, con un costo annuale superiore a 1 miliardo di dollari.

Un sistema completo di tracciabilità lungo l'intera catena di fornitura può aiutare le aziende a individuare e isolare rapidamente la causa di un'epidemia alimentare, consentendo una risposta tempestiva ed efficace per evitare la rimozione completa dei prodotti potenzialmente contaminati.

Inoltre, i processi di tracciabilità consentono alle aziende di individuare rapidamente i prodotti sospetti in qualsiasi punto della catena di fornitura, agevolando la rimozione dei prodotti ritirati per proteggere i consumatori e preservare la fiducia dei clienti. Una maggiore

visibilità nella catena di approvvigionamento e la consapevolezza dei consumatori sulla tracciabilità degli alimenti freschi dovrebbero anche contribuire a ripristinare la fiducia dei consumatori nella categoria di prodotto coinvolta, accelerando il ritorno alle abitudini di acquisto normali e ai livelli di vendita precedenti. Inoltre, i processi di tracciabilità aiutano le aziende a contenere costi imprevisti (come quelli legali, le multe, il rinnovo forzato dei contratti e la perdita di fedeltà dei clienti) e minimizzano gli impatti negativi sulle aziende coinvolte nella catena di fornitura e sui consumatori.

Ritorno sull'investimento

I vantaggi della visibilità nella catena di approvvigionamento influenzano il ritorno sull'investimento (ROI). I miglioramenti dei processi e le efficienze operative che si manifestano in tutta l'organizzazione devono essere considerati quando si valuta il ROI di qualsiasi iniziativa mirata all'implementazione o al miglioramento dei programmi di tracciabilità e visibilità nella catena di fornitura.

Uno strumento che può assistere in questo processo è il *Global Food Traceability Center* (GFTC) il quale aiuta le organizzazioni del settore ittico a valutare l'impatto finanziario, comprendendo costi e benefici, legato all'adozione della tracciabilità.

Il GFTC ha identificato che i benefici dalla tracciabilità possono derivare da:

- mercati e clienti nuovi ed ampliati;
- riduzione dei costi di responsabilità aziendale;
- efficienze di richiamo più elevate e costi di rilavorazione inferiori;
- riduzione degli scarti e dei restringimenti del prodotto;
- scambio di dati più affidabile e facilmente accessibile con i partner;
- costi di adeguamento normativo più rapidi e inferiori.

2.2.3 Tracciabilità dei prodotti sfusi

La tracciabilità dei prodotti alimentari sfusi è una sfida unica nel settore alimentare. Le filiere dei prodotti alimentari sfusi possono essere descritte in modo più accurato come un sistema alimentare, a causa della complessa rete che si crea intorno a questi prodotti. Il loro percorso, dalla fattoria o dal fornitore fino al punto finale, non segue un percorso lineare. Spesso, il prodotto viene mescolato con prodotti simili provenienti da diversi lotti o lotti o viene completamente unito ad altri prodotti.

I prodotti alimentari sfusi sono particolarmente vulnerabili alla contaminazione incrociata accidentale o alla miscelazione durante lo stoccaggio. Ciò è dovuto sia alla naturale composizione di tali prodotti che al costo e alla praticità di prevenire completamente il mescolamento. Le materie prime vengono raccolte e gli ingredienti o i prodotti sono generalmente conservati in contenitori come serbatoi, bidoni, silos e altri recipienti appropriati. Durante il processo di produzione, i prodotti crudi o parzialmente trasformati possono subire modifiche tramite processi chimici e/o microbiologici, come la pastorizzazione, la sterilizzazione, la concentrazione, la diluizione e la fermentazione. Queste modifiche richiedono la generazione di codici di lotto.

Il monitoraggio di queste informazioni diventa essenziale durante il processo di produzione, che spesso non è continuo e richiede l'immagazzinamento del prodotto intermedio. Anche quando le fasi di lavorazione sono ben definite e i numeri di lotto sono chiaramente identificati, mantenere questa distinzione durante lo stoccaggio può essere difficile.

La tracciabilità dei prodotti alimentari sfusi è definita come la capacità di determinare ogni porzione della composizione di un prodotto grezzo, intermedio o finito in termini di input e output nel sistema. Storicamente, la necessità di identificare queste proporzioni di prodotti alimentari era principalmente guidata dalla gestione dei richiami alimentari. Tuttavia, nel contesto del sistema alimentare statunitense in evoluzione, la trasparenza è emersa come un fattore chiave per le aziende alimentari. L'importanza di mantenere un programma completo di tracciabilità per tutte le linee di prodotto è passata da essere una misura difensiva a diventare una misura opportunistica per migliorare l'efficienza delle aziende alimentari.

Per ottenere una tracciabilità completa dei prodotti sfusi all'interno del sistema alimentare, è necessario registrare tutte le informazioni rilevanti, compresi, ma non limitandosi ai numeri di lotto, i pesi, le condizioni di manipolazione e l'elaborazione delle informazioni.

Gestione della tracciabilità dei prodotti sfusi

Il processo fondamentale per spostare un singolo prodotto dall'ingresso all'uscita coinvolge numerosi punti di contatto che richiedono la raccolta di dati in ogni fase. Tuttavia, per i prodotti sfusi, questi passaggi intermedi possono rappresentare una sfida maggiore da monitorare. ciò è dovuto al fatto che, mentre i prodotti vengono mescolati o separati, il produttore deve creare nuove identificazioni o informazioni univoche, come numeri di lotto o altri identificatori, per tracciare i prodotti e i loro componenti avanti e indietro. La gestione di questi passaggi intermedi dipende dall'approccio specifico del produttore alla produzione e alla lavorazione dei propri prodotti.

Esistono due approcci principali alla tracciabilità utilizzati per verificare le affermazioni di sostenibilità e altre pratiche di produzione di prodotti alimentari in varie fasi della catena di approvvigionamento. Questi approcci variano a seconda del grado di controllo richiesto nella gestione dei prodotti. I due schemi di gestione della certificazione includono:

- *Segregazione del prodotto*: questo approccio prevede la separazione fisica dei prodotti certificati da quelli non certificati in ogni fase della catena di fornitura;
- *Bilancio di massa*: questo approccio consente la presenza di prodotti certificati e non certificati all'interno della catena del valore, a condizione che la percentuale di prodotti certificati superi una soglia specifica (ad esempio, 80% certificati vs. 20% non certificati).

L'applicazione di questi approcci di tracciabilità varia in base alla natura dei prodotti monitorati e ad altri fattori, tra cui i costi e le migliori pratiche. Il costo svolge un ruolo chiave per la maggior parte dei fornitori e delle aziende all'interno della catena del valore, soprattutto quando si tratta di sistemi di bilancio di massa altamente complessi. Maggiore è il controllo richiesto per tracciare un prodotto attraverso il sistema, e maggiore sarà la complessità della tracciabilità, con conseguente aumento dei costi.

Tecnologie impiegate

Negli ultimi anni, la ricerca sulla tracciabilità alimentare ha ampliato il proprio campo di interesse, andando oltre la mera sicurezza alimentare e includendo aspetti legati all'efficienza operativa, come la freschezza dei prodotti, la gestione dell'inventario e la riduzione degli sprechi. Tecnologie abilitanti che consentono operazioni più veloci e precise e un'identificazione più economica dei prodotti alimentari sfusi includono dispositivi RFID, sensori connessi in rete, codici a barre lineari e bidimensionali e sistemi diagnostici avanzati. L'utilizzo di questi strumenti in combinazione con dispositivi informatici mobili connessi a Internet consente di identificare, acquisire e condividere rapidamente informazioni su lotti e quantità di prodotto.

Tuttavia, vi sono ancora sfide nella raccolta e condivisione delle informazioni in tutto il sistema. Le materie prime agricole spesso attraversano catene di fornitura lunghe e complesse che si estendono su vaste aree geografiche, attraversano molteplici confini e coinvolgono aziende di varie dimensioni e livelli di tecnologia. Il complesso ambiente normativo, la copertura Internet incompleta e le differenze negli investimenti tecnologici creano un terreno fertile per le sfide nella gestione della tracciabilità degli alimenti sfusi. Nel futuro, ci si può aspettare che la riduzione dei costi, l'aumento dell'efficacia e una maggiore disponibilità delle soluzioni tecnologiche contribuiranno a superare queste sfide. Tuttavia, sarà compito dell'industria alimentare collaborare con i regolatori, gli acquirenti e i programmi di certificazione di terze parti per allineare i requisiti di tracciabilità e garantire un sistema efficace e affidabile.

Il futuro della tracciabilità dei prodotti sfusi

In passato, la tracciabilità dei prodotti alimentari sfusi era principalmente influenzata dalle esigenze legate ai cereali e alle materie prime. Tuttavia, il panorama del sistema alimentare sta subendo un rapido cambiamento, non solo negli Stati Uniti ma a livello globale, spingendo tutti gli attori della catena del valore ad adattarsi e a evolversi di conseguenza. Con il mutare delle esigenze e dei desideri dei consumatori, diventa essenziale sviluppare un sistema di tracciamento più accurato e uniforme in tutto il sistema.

Mentre si apportano miglioramenti all'aspetto operativo della tracciabilità, il futuro della tracciabilità dei prodotti sfusi si baserà sull'accelerato sviluppo di tecnologie per la comunicazione e la memorizzazione delle informazioni. L'integrazione di queste tecnologie emergenti sarà fondamentale per soddisfare la crescente necessità di tracciare in modo completo i prodotti alimentari lungo l'intera catena di approvvigionamento. Nonostante le sfide che questo panorama in evoluzione comporta, ci sono anche opportunità di crescita sia nel settore pubblico che in quello privato.

2.2.4 Tracciabilità della carne e pollame

Il tracciamento degli animali

La tracciabilità degli animali destinati alla produzione di carne e pollame ha inizio sin dalla fase di identificazione degli stessi. In passato, gli agricoltori segnavano gli animali per scopi di identificazione e di proprietà; nel corso degli ultimi decenni, l'identificazione degli animali è diventata cruciale anche per la tracciabilità in caso di epidemie e malattie animali. Ci sono molte ragioni per monitorare e tracciare gli animali, tra cui la creazione di prodotti a base di carne di valore aggiunto per scopi di marketing, la conformità alle normative, come l'etichettatura del paese di origine e i controlli di biosicurezza, che richiedono una corretta segregazione e separazione degli animali. Pertanto, i programmi di gestione efficiente per aziende agricole e attività di produzione e lavorazione alimentare includono spesso programmi di tracciabilità.

Tecnologie impiegate

L'*identificazione a radiofrequenza (RFID)* è una tecnologia disponibile da alcuni decenni, ma il suo utilizzo per scopi di tracciabilità è stato limitato a causa dei costi associati. Nell'industria della carne, ad esempio, la tecnologia RFID viene spesso implementata attraverso l'inserimento di dispositivi nei marchi auricolari dei bovini per tracciare l'identità e la posizione degli animali. L'estrazione delle informazioni dal dispositivo richiede un lettore RFID dedicato, e un sistema software di tracciabilità deve essere configurato per gestire tali informazioni. L'RFID è particolarmente utile perché consente di immagazzinare le informazioni

in un dispositivo integrato con limitata interazione umana. Sebbene i codici a barre possano contenere informazioni simili, essi sono molto meno costosi rispetto all'RFID. Attualmente, quest'ultimo è più adatto per prodotti di alto valore o grandi volumi. Tuttavia, è possibile che nel tempo i costi della tecnologia RFID diminuiscano, rendendola più accessibile all'industria.

Stanno emergendo ulteriori tecnologie all'avanguardia per migliorare ulteriormente la tracciabilità dei prodotti animali crudi. Ad esempio, le nuove tecnologie basate sul DNA possono essere utilizzate per consentire ai produttori di carne confezionata, trasformatori e rivenditori alimentari di tracciare l'origine dei prodotti a base di carne fino all'animale di origine. La tracciabilità della carne bovina potrebbe essere notevolmente migliorata prelevando campioni di DNA da ciascuna carcassa prima della trasformazione in carne macinata. Le informazioni relative a ciascun campione verrebbero inserite in un database. Successivamente, il prodotto finito a base di carne macinata potrebbe essere campionato per determinare gli animali specifici utilizzati per produrlo. Queste tecnologie di tracciabilità basate sul DNA potrebbero anche consentire ai partecipanti della catena di fornitura di autenticare e convalidare attributi come la naturalezza, la produzione biologica o la razza Angus dei prodotti a base di carne. Poiché l'identificatore univoco è già presente nella sequenza del DNA, questa tecnologia offre un grande potenziale per il tracciamento di esseri viventi, come gli animali. Le tecniche di sequenziamento e di codifica del DNA stanno diventando sempre più accessibili e possono variare da semplici test di sequenziamento a sequenziamenti completi del genoma, fornendo una vasta gamma di informazioni. Le tecnologie basate sul DNA offrono numerose opportunità nel campo del tracciamento e si prevede che il loro utilizzo continuerà a crescere man mano che la tecnologia migliora.

Infine, per quanto riguarda la distribuzione e la vendita dei prodotti, i codici a barre sono ampiamente utilizzati dall'industria. È possibile applicare etichette con codici a barre su una vasta gamma di prodotti per fornire informazioni sull'origine e sulla distribuzione dei prodotti stessi. Queste tecnologie hanno notevolmente migliorato la tracciabilità dei prodotti durante la distribuzione e la vendita, contribuendo a garantire una maggiore trasparenza nella catena di approvvigionamento.

2.3 La blockchain nella tracciabilità dei prodotti

2.3.1 Introduzione alla blockchain

La storia della tecnologia blockchain ha le sue radici nei settori della tecnologia finanziaria e dell'e-commerce, il che è un aspetto importante da considerare quando si applica questa tecnologia a diversi contesti. Questa tecnologia è emersa inizialmente come un esperimento nel campo dello scambio di valore, noto come *criptovalute*. Il termine *Blockchain* è stato introdotto per la prima volta da *Satoshi Nakamoto*, una figura pseudonima o un gruppo di individui, in un articolo del 2008 che concettualizzava blocchi di dati collegati tramite una catena crittografica in una rete. L'anno successivo, Nakamoto creò *Bitcoin*, basandosi su questo concetto, ed esso rimane ancora oggi la criptovaluta più riconosciuta e diffusa.

L'obiettivo principale di Nakamoto era quello di creare un sistema in cui le transazioni potessero avvenire senza l'intermediazione di entità consolidate, come le banche, al fine di rendere le transazioni stesse più trasparenti e meno suscettibili di corruzione. L'architettura alla base di queste criptovalute, la blockchain, sfrutta la potenza di una rete globale aperta e la combina con l'uso della crittografia per creare un ambiente sicuro e affidabile che permette lo scambio di valore o informazioni senza bisogno di intermediari di fiducia.

La tecnologia blockchain rappresenta una nuova iterazione di un concetto preesistente; infatti, i registri sono un elemento fondamentale delle attività aziendali, e la blockchain utilizza questa tecnologia per migliorare alcuni dei limiti dei registri tradizionali, riducendo la

dipendenza da entità esterne a favore della prova crittografica. Nel contesto della tracciabilità alimentare, le tecnologie blockchain sono considerate uno strumento potente per abilitare una tracciabilità completa e una maggiore trasparenza lungo tutta la catena di approvvigionamento. Con una blockchain distribuita su una rete condivisa, tutte le parti coinvolte nella catena di approvvigionamento possono accedere alle stesse informazioni sulla tracciabilità. Tuttavia, la vera rivoluzione sta nella possibilità che queste informazioni siano disponibili per tutti i segmenti della catena di fornitura, inclusi i consumatori finali.

Sebbene l'inizio della tecnologia blockchain si sia concentrata sulla creazione di criptovalute non istituzionali, la tecnologia è essenzialmente un registro con un vasto potenziale di funzionalità, a seconda dell'architettura. Per il nostro contesto, una transazione è semplicemente l'aggiunta o la modifica di informazioni sulla blockchain. Nella tracciabilità alimentare, ad esempio, un prodotto alimentare può subire una trasformazione interna e sarebbe, quindi, registrato sulla blockchain. Questo processo può essere ancora definito come una transazione, anche se non comporta un trasferimento di denaro.

L'utilizzo della blockchain nei sistemi di tracciabilità si basa su diverse caratteristiche chiave, tra cui la velocità di interrogazione del sistema, la capacità di garantire l'anonymato e la trasparenza simultaneamente, e la natura immutabile e condivisa del sistema. A differenza dei sistemi centralizzati, in cui esiste un singolo punto di vulnerabilità e opacità, la blockchain offre una soluzione che si basa sulla fiducia nella tecnologia anziché nel fornitore. La blockchain potrebbe consentire a diverse parti lungo la catena di fornitura di condividere dati in un registro condiviso che copre l'intera estensione del mercato, dal produttore al consumatore. Inoltre, le aziende possono inserire informazioni sulla tracciabilità senza rivelare dettagli proprietari o strategie di concorrenza.

Dal 2018, le soluzioni di catena di fornitura e tracciabilità basate su blockchain sono state principalmente oggetto di studi pilota e di implementazioni. Numerose aziende hanno cominciato ad esplorare l'utilizzo di piattaforme blockchain open source come *Ethereum* o *Hyperledger* di IBM per migliorare le proprie catene di fornitura. Alcuni di questi progetti pilota integrano anche altre tecnologie, come sensori *IoT* (*Internet of Things*).

2.3.2 Casi d'uso nella tracciabilità

I casi d'uso della blockchain nella tracciabilità alimentare sono sostanzialmente simili a quelli delle iniziative di tracciabilità in generale, il che spiega il grande interesse di molti leader del settore per questa tecnologia. Le iniziative e le tecnologie di tracciabilità alimentare si concentrano principalmente su cinque principali casi d'uso: la prevenzione delle frodi alimentari, la garanzia della sicurezza alimentare e dei richiami, la conformità normativa, le questioni sociali e la fornitura di informazioni ai consumatori.

La blockchain trova la sua massima utilità nella tracciabilità alimentare quando si tratta di prodotti alimentari, come, ad esempio, prodotti agricoli, e catene di approvvigionamento che sono frammentate, come nel caso dei prodotti ittici. Nei casi in cui le operazioni sono integrate verticalmente, l'utilità delle blockchain è limitata, poiché si possono sfruttare strumenti esistenti nella gestione dell'inventario per raggiungere gli obiettivi di tracciabilità.

Le frodi alimentari rappresentano una sfida per tutti i settori alimentari e richiedono miglioramenti nei metodi di rilevamento e maggiore tracciabilità delle informazioni. La *Food Fraud Initiative* dello Stato del Michigan definisce le frodi alimentari come "*adulterazione, etichettatura errata, manipolazione, superamento o contraffazione, furto, dirottamento, simulazione e contraffazione*". Spesso, le motivazioni economiche spingono all'alterazione di prodotti alimentari o delle informazioni lungo la catena di fornitura. La blockchain è stata vista come uno strumento potenziale per affrontare le frodi alimentari, poiché fornisce un registro distribuito, inalterabile e datato di ogni fase della catena di approvvigionamento, semplificando così l'audit per indagare su frodi alimentari.

La conformità normativa è un aspetto cruciale nella progettazione dei sistemi di tracciabilità alimentare, poiché la mancanza di conformità può comportare la vendita di prodotti non autorizzati, sanzioni e danni alla reputazione. Con l'aumento dei requisiti normativi per la tracciabilità alimentare su scala globale, la blockchain offre flessibilità ed efficacia nel soddisfare e anticipare tali regolamenti.

Un caso d'uso chiave affrontato direttamente dalla blockchain riguarda i richiami alimentari e la sicurezza. Le epidemie alimentari possono danneggiare gravemente la reputazione delle aziende e la blockchain fornisce un quadro decentralizzato ma unificato per tracciare il percorso dei prodotti alimentari lungo la catena di approvvigionamento. Questo richiede un'architettura dati ben progettata, che sia accessibile e conveniente per tutti i partner della catena di fornitura, promuovendo la condivisione delle responsabilità nella gestione dei dati, l'interoperabilità e la sicurezza dei dati.

La blockchain pubblica Ethereum possiede caratteristiche fondamentali che la rendono un'opzione attraente per il settore della tracciabilità alimentare. In queste blockchain pubbliche, le transazioni possono essere eseguite ad un costo molto ridotto, eliminando, così, gli ostacoli finanziari che spesso affliggono i fornitori, i quali hanno spesso margini di profitto limitati e risorse limitate per l'adozione di nuove tecnologie. Questo facilita notevolmente l'implementazione della tracciabilità.

Uno dei casi d'uso più promettenti nel settore delle startup basate su blockchain è il miglioramento delle informazioni fornite ai consumatori finali sui prodotti alimentari. Il movimento dell'etichettatura pulita e l'industria dei dati dimostrano che i consumatori sono sempre più interessati all'origine, alla produzione e alla catena di approvvigionamento dei prodotti che consumano. L'etichettatura pulita si impegna a garantire che certi ingredienti o additivi non siano stati utilizzati, ma soprattutto rappresenta una tattica di marketing che riflette il desiderio dei consumatori di avere accesso a informazioni dettagliate sui loro prodotti. Le etichette intelligenti rappresentano un altro esempio di come i consumatori desiderino sempre più informazioni. La blockchain, unificando la catena di approvvigionamento, offre alle aziende la possibilità di educare i propri consumatori sull'origine e sulla produzione dei loro prodotti.

2.3.3 Configurazioni della blockchain

I due principali tipi di ambienti blockchain, noti come *blockchain pubblici* e *blockchain privati*, hanno caratteristiche e priorità diverse. Le blockchain pubbliche, come *Bitcoin* ed *Ethereum*, sono aperti a chiunque abbia i token necessari per registrare transazioni sulla blockchain. Tuttavia, il consenso e le configurazioni private sono state sviluppate per adattarsi a esigenze diverse.

Le blockchain hanno priorità concorrenti che determinano l'efficienza e la privacy, e queste priorità variano a seconda del caso d'uso. In una blockchain pubblica veramente decentralizzata, l'accesso è aperto a tutti e le transazioni sono validate attraverso un processo di estrazione basato su token, come avviene con Bitcoin o Ether. Tuttavia, con l'espansione della blockchain, il tempo necessario per completare le transazioni aumenta notevolmente. Questa lentezza potrebbe non essere vantaggiosa per le catene di approvvigionamento, dove l'efficienza è fondamentale, e quindi sono state sviluppate le blockchain private o a consenso. Queste architetture conservano alcune delle caratteristiche desiderabili delle blockchain, come l'immutabilità, la data e l'ora di registrazione e la verificabilità. Tuttavia, rendendo la blockchain privata o semi-privata, si perdono alcune delle caratteristiche innovative della tecnologia, trasformandola essenzialmente in un tipo di database più tradizionale. Questo approccio può aumentare i costi, poiché richiede la gestione di nodi centrali, spesso controllati da un fornitore di servizi, e riduce la democratizzazione e la de-istituzionalizzazione promesse dalla blockchain.

Tuttavia, per le catene di approvvigionamento alimentare, non avere una rete blockchain completamente decentralizzata potrebbe non essere un problema critico. Molte delle attuali implementazioni della blockchain nel settore alimentare sono fornite da grandi aziende verticalmente integrate, che hanno le risorse per negoziare con i fornitori di servizi per coordinare, ospitare e gestire nodi di fiducia nella rete. Queste aziende possono anche collaborare direttamente con i partner della catena di fornitura per garantire l'adozione efficace delle migliori pratiche e tecnologie.

CAPITOLO 3

Descrizione generale del progetto

Nel terzo capitolo si presenterà una panoramica del progetto di questa tesi. Si inizierà con una breve introduzione relativa al suo scopo generale. Successivamente, si approfondiranno i dettagli relativi agli strumenti software utilizzati nel progetto, tra cui Unity, Vuforia Engine, Polycam, Blender, Vuforia Model Target Generator, MongoDB, Google Earth Studio e OpenWeather.

In seguito, si esplorerà nel dettaglio la stazione IoT situata nel vigneto della cantina Strappelli, che è stata messa a disposizione per il monitoraggio dei dati relativi al vigneto. In aggiunta, si fornirà una breve illustrazione del funzionamento dell'applicazione, descrivendo tutte le fasi che vanno dal lancio del menu principale fino alla visualizzazione dei dati relativi al vigneto associato al prodotto vitivinicolo riconosciuto dall'applicazione. Il tutto verrà illustrato fornendo degli screenshot per facilitare la comprensione di quello che verrà descritto.

3.1 Premessa

In questo capitolo, si esplorerà in dettaglio la creazione di un'app mobile per dispositivi Android. Questa app avrà la capacità di riconoscere automaticamente prodotti vitivinici attraverso la fotocamera dello smartphone, mettendo particolare enfasi sulle bottiglie di vino. Un elemento chiave è che il riconoscimento non si baserà sui tradizionali codici QR, ma farà uso di modelli tridimensionali degli oggetti precedentemente caricati all'interno dell'applicazione in modo tale da non alterare le etichette delle bottiglie per far funzionare l'applicazione. Una volta completata con successo la fase di riconoscimento, l'app fornirà all'utente informazioni dettagliate sul prodotto identificato.

3.2 Scopo generale dell'applicazione

Lo scopo dell'applicazione è quello di fornire informazioni sul vigneto di provenienza delle bottiglie di vino riconosciute, combinando i dati dell'azienda Trace Technologies con dati satellitari provenienti dall'API di OpenWeather. La struttura dei dati utilizzati sarà descritta nel quarto capitolo.

Una volta effettuato il riconoscimento, l'app sarà suddivisa in tre sezioni principali:

- *Osserva:* in questa sezione, gli utenti avranno a disposizione una ricca fonte di informazioni, sia testuali che multimediali, relative al prodotto vitivinicolo appena identificato. Sarà possibile esplorare il nome della cantina produttrice, avere una breve descrizione

del vino, nonché dettagli informativi sul luogo in cui esso è stato prodotto. Inoltre, per offrire un’esperienza visiva coinvolgente, l’app presenterà un breve video che fornirà una vista panoramica dall’alto, sfruttando immagini satellitari ottenute da Google Earth Studio, del luogo di produzione del prodotto vitivinicolo.

- *Ascolta*: questa sezione opera in modo simile a un karaoke, ma con una differenza fondamentale: gli utenti potranno godere di una recensione testuale del vino, redatta da un sommelier esperto, sincronizzata in tempo reale con l’audio fornito dallo stesso sommelier. Questa caratteristica rappresenta un notevole miglioramento dell’esperienza dell’utente, arricchendo l’interazione con l’app in modo significativo.
- *Racconta*: l’ultima sezione è dedicata alla visualizzazione dello stato di salute del vigneto di provenienza delle bottiglie di vino. Questa sezione si divide in due parti essenziali: la prima parte fornisce dati provenienti dalla stazione IoT di Trace Technologies, inclusi parametri come temperatura e umidità ambientale, quantità giornaliera di pioggia, temperatura delle foglie e umidità delle foglie del vigneto. La seconda parte presenta lo stato di salute del vigneto utilizzando dati ottenuti dall’API OpenWeather.

3.3 Strumenti software utilizzati

3.3.1 Unity

Unity (Figura 3.1) è una delle piattaforme di sviluppo di giochi e simulazioni più potenti e versatili disponibili oggi. È un ambiente di sviluppo integrato (IDE) che offre agli sviluppatori una vasta gamma di strumenti e risorse per creare giochi, applicazioni interattive e simulazioni per una vasta gamma di piattaforme, compresi PC, console, dispositivi mobili, realtà virtuale (VR) e realtà aumentata (AR).



Figura 3.1: Logo di Unity

Storia di Unity

La storia di Unity risale al 2005, quando fu sviluppato da David Helgason, Joachim Ante e Nicholas Francis. Da allora, Unity Technologies ha continuato a crescere ed evolversi, diventando uno dei leader nel settore dello sviluppo di giochi e applicazioni 3D. Attualmente, Unity è utilizzato da milioni di sviluppatori in tutto il mondo, dai principianti agli sviluppatori esperti.

Aspetti distintivi e funzionalità

Uno degli aspetti distintivi di Unity è la sua flessibilità. Esso supporta una vasta gamma di linguaggi di programmazione, tra cui C e JavaScript, consentendo agli sviluppatori di utilizzare il linguaggio con cui si sentono più a loro agio. Questa flessibilità rende Unity un’ottima scelta per sviluppatori con diverse competenze ed esperienze.

Unity offre una serie di funzionalità e strumenti avanzati, tra cui:

- *Grafica avanzata*: Unity supporta una grafica di alta qualità con rendering in tempo reale, illuminazione globale, riflessi, ombre dinamiche e supporto per il ray tracing. Gli sviluppatori possono creare mondi virtuali incredibilmente dettagliati e realistici.
- *Fisica realistica*: Unity include un motore fisico avanzato che consente agli sviluppatori di simulare il comportamento realistico degli oggetti, dalla caduta degli oggetti alla dinamica dei veicoli.
- *Animazione*: gli sviluppatori possono creare animazioni complesse per personaggi e oggetti utilizzando il sistema di animazione di Unity, che supporta la blend tree, l'animazione retargeting e altro ancora.
- *Intelligenza artificiale*: Unity offre strumenti per la creazione di comportamenti di Intelligenza Artificiale (IA) per personaggi non giocanti e avversari controllati dall'IA.
- *Realtà virtuale e aumentata*: Unity è una scelta popolare per lo sviluppo di applicazioni VR e AR, con supporto integrato per numerosi dispositivi e librerie di sviluppo.
- *Multi-piattaforma*: gli sviluppatori possono creare giochi e applicazioni per una vasta gamma di piattaforme, compresi Windows, macOS, Linux, iOS, Android, PlayStation, Xbox e molte altre.
- *Asset Store*: Unity dispone di un Asset Store che offre una vasta selezione di risorse pronte all'uso, come modelli 3D, suoni, effetti speciali e plugin, che possono accelerare notevolmente il processo di sviluppo.
- *Community e supporto*: Unity ha una vasta comunità di sviluppatori attivi, forum di supporto e una documentazione dettagliata che aiutano gli sviluppatori a risolvere i problemi e a imparare nuove tecniche.

Inoltre, Unity è utilizzato in una varietà di settori oltre ai videogiochi, tra cui l'architettura, la simulazione, la formazione, la medicina e l'arte interattiva. La sua versatilità e la sua capacità di adattarsi a una vasta gamma di progetti lo rendono uno strumento eccezionale per chiunque desideri creare esperienze digitali coinvolgenti e interattive.

3.3.2 Vuforia Engine

Vuforia Engine (Figura 3.2) è una potente piattaforma di realtà aumentata (AR) sviluppata da PTC (*Parametric Technology Corporation*) che consente agli sviluppatori di creare applicazioni AR immersive e interattive per una vasta gamma di dispositivi e piattaforme. Questo strumento è stato utilizzato in molte applicazioni, tra cui giochi, applicazioni industriali, educative e di marketing, per portare il mondo fisico e il digitale insieme in modo innovativo.



Figura 3.2: Logo di Vuforia Engine

Caratteristiche di Vuforia Engine

Le caratteristiche principali di Vuforia Engine sono:

- *Rilevamento degli oggetti e dei marker*: una delle caratteristiche chiave di Vuforia Engine è la sua capacità di rilevare oggetti fisici o marker grafici nel mondo reale e di tracciare la loro posizione e orientamento. Questo avviene attraverso la visione computerizzata e l'analisi dell'immagine catturata dalla fotocamera del dispositivo AR. I marker possono essere stampati su carta o inseriti in oggetti reali.
- *Tracciamento dei movimenti*: Vuforia Engine offre anche funzionalità di tracciamento dei movimenti, consentendo agli oggetti AR di rimanere ancorati al loro punto di riferimento anche quando l'utente si sposta. Questo è fondamentale per garantire un'esperienza AR fluida e realistica.
- *Riconoscimento dell'ambiente*: oltre al rilevamento di marker, Vuforia Engine può riconoscere l'ambiente circostante, consentendo agli oggetti AR di interagire con gli elementi reali dell'ambiente. Ad esempio, un'applicazione AR può posizionare un oggetto virtuale su un tavolo o farlo interagire con una parete.
- *Rendering 3D*: una volta che Vuforia Engine ha acquisito e tracciato le informazioni sull'ambiente e sugli oggetti, può visualizzare oggetti 3D virtuali nell'ambiente del mondo reale. Questo processo di rendering 3D viene effettuato in tempo reale per garantire che gli oggetti AR si integrino perfettamente nell'ambiente.
- *Interazione utente*: Vuforia Engine offre anche supporto per l'interazione utente. Gli utenti possono interagire con gli oggetti AR attraverso gesti, tocchi e input vocali, consentendo loro di manipolare oggetti virtuali e di ricevere risposte dall'applicazione.
- *Multi-piattaforma*: Vuforia Engine è compatibile con una vasta gamma di dispositivi, tra cui smartphone, tablet, occhiali AR e HoloLens di Microsoft. Questa flessibilità consente agli sviluppatori di creare applicazioni AR che raggiungono un pubblico ampio e diversificato.
- *Sicurezza e privacy*: Vuforia Engine si preoccupa della sicurezza e della privacy degli utenti, garantendo che le applicazioni AR rispettino le normative sulla protezione dei dati e consentano agli utenti di controllare le autorizzazioni per l'accesso alla fotocamera e ad altre informazioni personali.
- *Supporto per lo sviluppatore*: Vuforia Engine offre una serie di strumenti e risorse per gli sviluppatori, tra cui un'ampia documentazione, esempi di codice, un'interfaccia di sviluppo intuitiva e un'attiva community online. Ciò facilita la creazione di applicazioni AR di alta qualità.

3.3.3 Polycam

Polycam (Figura 3.3) è un'applicazione mobile in grado di generare un modello tridimensionale a partire da una sequenza di foto scattate con il proprio smartphone.



Figura 3.3: Logo di Polycam

Esistono quattro modi diversi per creare una scansione in Polycam. questi sono descritti di seguito:

- *Photo Mode*: è di solito la scelta migliore per gli oggetti che richiedono una grande precisione e un grande dettaglio, o quando non si ha accesso a un sensore LiDAR. Questo metodo funziona scattando una sequenza di fotografie standard dell'oggetto che vengono poi caricate in un server più potente che crea la ricostruzione tridimensionale con la massima precisione possibile.
- *LiDAR Mode*: è una modalità comoda per l'acquisizione di spazi o quando si desidera effettuare una scansione rapidamente. Il metodo utilizza i dati di profondità forniti dal sensore LiDAR che permette di generare una ricostruzione tridimensionale dell'ambiente direttamente sul dispositivo. Questo approccio sfrutta le capacità di misurazione precisa del sensore LiDAR per ottenere risultati veloci e affidabili nella creazione di modelli 3D.
- *Room Mode*: questa modalità consente di generare planimetrie professionali in 3D e 2D e di misurare istantaneamente gli spazi interni.
- *360 Mode*: questa modalità permette di scattare foto panoramiche a 360° con l'ausilio dell'Intelligenza Artificiale.

3.3.4 Blender

Blender (Figura 3.4) è un software di grafica 3D gratuito ed open source noto per la sua potenza e flessibilità. È utilizzato per la modellazione, l'animazione, la simulazione, il rendering, la composizione e il montaggio video.



Figura 3.4: Logo di Blender

Storia di Blender

La storia di Blender inizia nel 1988 quando Ton Roosendaal, un programmatore olandese, inizia a sviluppare un'applicazione di grafica 3D chiamata "Traces" su un computer Amiga. Nel 1995, Roosendaal fondò la società Not a Number (NaN) per sviluppare e distribuire Blender. Tuttavia, a causa di problemi finanziari, NaN dichiarò bancarotta nel 2002.

La comunità degli utenti di Blender, desiderosa di non perdere questa preziosa risorsa, si è mobilitata e ha raccolto fondi per acquistare il codice sorgente di Blender. Nel 2002, Ton Roosendaal ha accettato di rilasciare il codice sorgente sotto una licenza open source, dando vita alla Blender Foundation. Da allora, Blender ha continuato a crescere e a evolversi grazie al contributo di una vasta comunità di sviluppatori, artisti e appassionati.

Caratteristiche di Blender

Blender è noto per la sua interfaccia utente unica, che può sembrare complessa ai principianti ma offre un'ampia gamma di strumenti e funzionalità per gli utenti avanzati.

Ecco alcuni aspetti chiave di Blender:

- *Modellazione 3D*: Blender consente agli utenti di creare oggetti tridimensionali utilizzando una varietà di strumenti, tra cui mesh, curve e superfici. È possibile modellare oggetti da zero o importare modelli da altre fonti.
- *Animazione*: Blender supporta l'animazione tramite keyframe, rigging e skinning. Gli utenti possono creare animazioni fluide per personaggi, oggetti e scene.
- *Simulazione*: il software offre funzionalità di simulazione per la dinamica dei fluidi, il fumo, il tessuto e molte altre simulazioni fisiche. Queste caratteristiche sono utili per la creazione di effetti realistici.
- *Rendering*: Blender offre un motore di rendering integrato chiamato Cycles, che consente di generare immagini fotoreali. È anche possibile utilizzare il motore di rendering Eevee per l'anteprima in tempo reale.
- *Composizione e montaggio video*: Blender non è solo uno strumento di modellazione e animazione, ma può anche essere utilizzato per la composizione di scene, il montaggio video e l'aggiunta di effetti speciali.
- *Scripting e personalizzazione*: Blender supporta lo scripting Python, il che significa che gli utenti avanzati possono estendere le funzionalità del software creando script personalizzati.
- *Rendering distribuito*: Blender supporta il rendering distribuito su più computer, il che accelera notevolmente il processo di rendering per progetti complessi.
- *Comunità e risorse*: Blender ha una comunità molto attiva che offre tutorial, risorse gratuite e supporto online attraverso forum e social media.

3.3.5 Vuforia Model Target Generator

Vuforia Model Target Generator è un software di generazione di modelli tridimensionali. In particolare, esso prende in input un modello tridimensionale e produce in output un modello target da importare all'interno dell'applicazione Unity per permettere al motore grafico Vuforia Engine di effettuare il riconoscimento automatico delle immagini.

Il programma è in grado anche di creare un database formato da più modelli di oggetti target su cui poi effettuare un addestramento di un modello di Machine Learning per il riconoscimento automatico di target multipli in una scena Unity rappresentata dalle immagini acquisite dalla fotocamera dello smartphone.

3.3.6 MongoDB

MongoDB (Figura 3.5) è un sistema di gestione di database orientato ai documenti (DBMS) di tipo NoSQL ampiamente utilizzato per l'archiviazione e il recupero di dati. La sua struttura flessibile e scalabile lo rende popolare per una vasta gamma di applicazioni, dalle piccole app web alle grandi applicazioni aziendali.



Figura 3.5: Logo di MongoDB

Architettura di MongoDB

Ecco una descrizione dell'architettura di MongoDB:

- *Modello di dati basato su documenti*: MongoDB archivia i dati in documenti JSON-like chiamati *BSON* (*Binary JSON*), che rappresentano entità dei dati. I documenti sono raggruppati in collezioni, che sono l'equivalente delle tabelle in un database relazionale.
- *Database*: un'istanza di MongoDB può contenere uno o più database, ognuno dei quali può avere diverse collezioni di documenti. Ogni database ha un proprio set di autorizzazioni e credenziali di accesso.
- *Collezioni*: le collezioni sono gruppi di documenti simili o correlati all'interno di un database. Ad esempio, in un'applicazione di e-commerce, potrebbe esserci una collezione per i prodotti e un'altra per gli ordini.
- *Documenti*: i documenti MongoDB sono oggetti JSON-like composti da coppie chiave-valore. Questi documenti possono contenere campi di dati di tipo diverso, tra cui stringhe, numeri, array e altri documenti nidificati.

Principali caratteristiche e funzionalità di MongoDB

Le principali caratteristiche e funzionalità di MongoDB sono le seguenti:

- *Scalabilità*: MongoDB è altamente scalabile e supporta sia la scalabilità orizzontale (sharding) che quella verticale (ridimensionamento dei server). Ciò consente ad esso di gestire grandi volumi di dati e di crescere con le esigenze dell'applicazione.
- *Alta disponibilità*: MongoDB offre meccanismi di replica che consentono di creare repliche dei dati in più server per garantire l'alta disponibilità e la tolleranza ai guasti.
- *Flessibilità del modello di dati*: la flessibilità del modello di dati JSON-like consente di gestire dati semi-strutturati o completamente non strutturati senza dover seguire uno schema rigido.
- *Query potenti*: MongoDB supporta query complesse e indicizzazione per il recupero efficiente dei dati. Le query possono essere eseguite utilizzando il linguaggio di query di MongoDB.
- *Aggregazione*: MongoDB offre un framework di aggregazione che consente di eseguire operazioni di aggregazione avanzate sui dati, come raggruppamenti, somme, medie e altro ancora.
- *Geospatial*: MongoDB ha funzionalità native per la gestione di dati geospaziali, consentendo di archiviare e interrogare dati basati sulla posizione geografica.
- *Replica set*: questa funzionalità consente di garantire l'alta disponibilità dei dati attraverso la replicazione sincrona o asincrona su più server.
- *Sharding*: MongoDB supporta lo sharding, consentendo la distribuzione dei dati su cluster di server per gestire volumi di dati molto grandi e carichi di lavoro ad alta intensità di lettura/scrittura.
- *Sicurezza*: MongoDB offre funzionalità di autenticazione, autorizzazione e crittografia dei dati per garantire la sicurezza degli stessi.

- *Community e supporto:* MongoDB ha una vasta comunità di utenti e sviluppatori attivi, oltre a un'offerta di supporto professionale per le aziende che richiedono assistenza.

MongoDB è una scelta popolare per applicazioni che richiedono flessibilità, scalabilità e gestione di dati semi-strutturati. La sua capacità di adattarsi a una vasta gamma di casi d'uso lo rende una soluzione potente per molte applicazioni moderne.

3.3.7 Google Earth Studio

Google Earth Studio è una potente e innovativa applicazione di animazione e visualizzazione geo-spatiale sviluppata da Google. Questo strumento è stato creato per consentire agli utenti di creare animazioni e video basati su mappe e immagini satellitari provenienti da Google Earth e Google Maps. Ecco una descrizione dettagliata di Google Earth Studio:

- *Interfaccia utente intuitiva:* Google Earth Studio offre un'interfaccia utente intuitiva e basata su browser che è facilmente accessibile da qualsiasi dispositivo connesso ad Internet. Gli utenti possono iniziare a creare animazioni geospaziali senza dover scaricare o installare alcun software aggiuntivo.
- *Visualizzazione geospaziale avanzata:* l'applicazione sfrutta i dati di Google Earth e Google Maps per offrire agli utenti una visualizzazione dettagliata e interattiva del mondo. È possibile esplorare mappe, immagini satellitari, modelli 3D e strati di dati geospaziali.
- *Creazione di animazioni personalizzate:* Google Earth Studio consente agli utenti di creare animazioni geospaziali personalizzate utilizzando una vasta gamma di strumenti e funzionalità, come, ad esempio:
 - *Navigazione fluide:* gli utenti possono definire percorsi di volo e itinerari sulla mappa, consentendo di spostarsi da un punto all'altro con transizioni fluide;
 - *Aggiunta di pin ed etichette:* è possibile inserire marcatori, punti di interesse ed etichette per evidenziare luoghi specifici sulla mappa;
 - *Controllo del tempo:* gli utenti possono specificare la durata dell'animazione e controllare la velocità di avanzamento temporale;
 - *Zoom e rotazione:* è possibile applicare zoom in avanti/indietro ed effettuare rotazioni per modificare la prospettiva della mappa;
 - *Personalizzazione dei colori e degli stili:* gli utenti possono personalizzare gli stili di mappa, i colori di sfondo e gli effetti visivi per ottenere l'aspetto desiderato;
 - *Importazione di dati:* è possibile importare dati esterni, come dati demografici o informazioni geografiche, e di sovrapporli alla mappa.
- *Rendering di alta qualità:* Google Earth Studio offre strumenti di rendering di alta qualità che consentono agli utenti di esportare le loro animazioni in formato video o immagine con risoluzioni fino a 4K. Questo consente la creazione di contenuti visivi di alta definizione per presentazioni, progetti cinematografici, video tutorial e altro ancora.
- *Integrazione con altri strumenti:* Google Earth Studio offre un'integrazione diretta con altri strumenti di creazione video e grafica, come Adobe After Effects. Questo permette agli utenti di arricchire ulteriormente le loro animazioni aggiungendo effetti speciali, titoli, transizioni e altro.
- *Accesso gratuito e a pagamento:* l'applicazione è disponibile con un livello gratuito che consente di creare progetti con alcune limitazioni, ma offre anche piani a pagamento per accedere a funzionalità avanzate e ulteriori risorse.

3.3.8 OpenWeather



Figura 3.6: Logo di OpenWeather

OpenWeather (Figura 3.6) è un servizio di previsioni meteorologiche e di dati meteorologici in tempo reale che fornisce informazioni dettagliate sulle condizioni meteorologiche in tutto il mondo. L'API di OpenWeather è ampiamente utilizzata da sviluppatori di applicazioni, siti web e servizi per integrare dati meteorologici accurati nelle loro applicazioni. Essa presenta le seguenti caratteristiche:

- *Dati meteorologici attuali:* l'API di OpenWeather fornisce informazioni aggiornate in tempo reale sulle condizioni meteorologiche, inclusi dati come temperatura, umidità, velocità del vento, pressione atmosferica, visibilità, e altro ancora. Questi dati sono fondamentali per fornire agli utenti informazioni immediate sul clima nell'area di loro interesse.
- *Previsioni meteorologiche:* OpenWeather offre previsioni meteorologiche per un periodo di tempo specifico, che può variare da poche ore a diversi giorni. Le previsioni includono dettagli come temperatura massima e minima, condizioni del cielo, probabilità di precipitazioni, vento e altro. Gli sviluppatori possono specificare il periodo di previsione richiesto.
- *Dati storici:* l'API consente l'accesso ai dati meteorologici storici, consentendo agli utenti di recuperare informazioni sulle condizioni meteorologiche passate in una determinata località. Questo può essere utile per scopi di analisi e riferimento storico.
- *Dati geospaziali:* OpenWeather fornisce dati basati su coordinate geografiche, consentendo agli sviluppatori di ottenere informazioni meteorologiche specifiche per una posizione geografica precisa. Questo è utile per applicazioni mobili e servizi basati sulla posizione.
- *Immagini radar e satellitari:* l'API offre accesso a immagini radar e satellitari in tempo reale che mostrano il movimento delle nuvole e delle precipitazioni. Queste immagini possono essere utilizzate per visualizzare i modelli meteorologici e le condizioni meteo in una regione specifica.
- *Notifiche e allarmi:* OpenWeather consente agli sviluppatori di impostare notifiche e allarmi basati su determinate condizioni meteorologiche, come avvisi per temporali imminenti o condizioni di gelo. Questo è utile per informare gli utenti in modo proattivo sulle situazioni meteorologiche pericolose.
- *Linguaggi e unità di misura:* l'API di OpenWeather supporta una vasta gamma di lingue e consente agli sviluppatori di specificare le unità di misura preferite (ad esempio, Celsius o Fahrenheit per la temperatura, metri o miglia per la distanza).
- *Accesso gratuito e a pagamento:* OpenWeather offre un livello gratuito di accesso all'API con limiti di utilizzo, ma offre anche piani a pagamento per un accesso più completo, con funzionalità aggiuntive e un maggiore numero di chiamate API.

L'API di OpenWeather è utilizzata in una vasta gamma di applicazioni, comprese app mobili, siti web, stazioni meteorologiche personali, sistemi di navigazione, e molto altro ancora. Offre dati accurati e aggiornati, ed è una risorsa preziosa per chiunque abbia bisogno di informazioni meteorologiche affidabili e aggiornate.

3.4 Strumentazione IoT utilizzata

Come già anticipato in precedenza, per valutare lo stato di salute del vigneto in tempo reale, l'applicazione utilizza sia dati provenienti dall'API OperWeather sia dati provenienti da una stazione IoT di proprietà dell'azienda Trace Technologies.

In Figura 3.7 viene mostrata una porzione della stazione IoT impiegata nel campo dell'azienda vitivinicola Strappelli situata in provincia di Teramo nel comune di Torano Nuovo.



Figura 3.7: La stazione IoT presente nel campo dell'azienda vitivinicola Strappelli

L'azienda Trace Technologies ha posizionato la stazione IoT nelle vicinanze della vite monitorando i seguenti valori atmosferici e valori utili con lo scopo di stabilire lo stato di salute del vigneto:

- temperatura ed umidità atmosferica;
- millimetri di pioggia (giornalieri, del giorno precedente, istantanei e totali) misurati tramite un pluviometro posizionato nella parte superiore della stazione (è visibile un contenitore cilindrico nella Figura 3.7);
- temperatura ed umidità della foglia (misurate tramite un sensore mostrato in Figura 3.8);
- temperatura ed umidità del terreno (misurate tramite un sensore inserito all'interno del terreno).

Per rendere il progetto sostenibile dal punto di vista ambientale, la stazione IoT è alimentata da un pannello fotovoltaico che durante il giorno fornisce l'energia necessaria alla stazione mentre l'energia che non viene impiegata durante le ore diurne è immagazzinata all'interno di una batteria che alimenta il sistema durante le ore notturne.

La stazione possiede, inoltre, un modulo 3G per la connessione ad Internet tramite una scheda SIM posta al suo interno.



Figura 3.8: Sensore di temperatura ed umidità della foglia

I dati raccolti dalla stazione vengono memorizzati localmente all'interno di una scheda SD per poi essere caricati su una piattaforma cloud che permette l'accesso dei dati da remoto tramite un'apposita API.

3.5 Funzionamento dell'applicazione

Al momento dell'avvio dell'applicazione viene visualizzato un menu che contiene tre pulsanti. Questi ultimi consentono all'utente di selezionare quale delle tre sezioni disponibili (Osserva, Ascolta, Racconta) desidera esplorare dopo che il riconoscimento della bottiglia è stato completato. In Figura 3.9 è presente uno screenshot del menù appena descritto.



Figura 3.9: Menu principale dell'applicazione

Dopo aver scelto la sezione di interesse, viene mostrata all'utente una schermata composta da un pulsante che permette di accedere alla scena successiva incentrata nell'attivazione della camera AR per il riconoscimento automatico del prodotto vitivinicolo. In Figura 3.10, è mostrato il pulsante che attende il tocco dell'utente prima di iniziare il riconoscimento del prodotto.



Figura 3.10: Pulsante che chiede all'utente di iniziare il riconoscimento del prodotto vitivinicolo

Dopo aver premuto uno dei pulsanti, l'applicazione attiva la fotocamera in modalità AR e avvia la ricerca della bottiglia di vino nell'ambiente inquadrato. L'obiettivo è individuare la bottiglia e identificare l'azienda produttrice del vino. Successivamente, l'applicazione mostra le informazioni disponibili sull'azienda, fornite da Trace Technologies. In Figura 3.11 è mostrato uno screenshot della scena Unity.

Per scopi dimostrativi, è stato incluso nell'applicazione un pulsante denominato "salta", il quale consente di procedere alla scena successiva anche se la bottiglia di vino non è stata riconosciuta. In situazioni in cui l'utente non possiede effettivamente la bottiglia di vino, verrà impostata una variabile che registra il tag della bottiglia riconosciuta con il valore "Strappelli".

Una volta completato il processo di riconoscimento della bottiglia, l'applicazione Unity passa ad una nuova scena. In quest'ultima, la fotocamera rimane attiva ma l'algoritmo di riconoscimento degli oggetti viene disattivato per risparmiare risorse. Successivamente, viene mostrata una finestra a comparsa dall'alto che chiede all'utente di selezionare l'annata della bottiglia che possiede. Questa scelta implementativa è stata necessaria poiché Vuforia Engine non è sempre in grado di determinare con precisione l'annata della bottiglia dall'etichetta, poiché quest'ultima potrebbe non essere sempre chiaramente visibile.

In Figura 3.12 è mostrata la finestra scorrevole appena descritta.



Figura 3.11: Scena AR dedicata al riconoscimento del prodotto vitivinicolo

Selezionata l'annata, l'applicazione Unity carica l'ultima scena che contiene tutte le sezioni menzionate in precedenza (Osserva, Ascolta e Racconta) mostrate, rispettivamente, nelle Figure 3.13, 3.14 e 3.15.



Figura 3.12: Finestra per la selezione dell'annata del prodotto vitivinicolo



Figura 3.13: Screenshot della sezione Osserva



Figura 3.14: Screenshot della sezione Ascolta



Figura 3.15: Screenshot della sezione Racconta

CAPITOLO 4

Progettazione del sistema

In questo capitolo, si esplorerà in dettaglio la progettazione dell'applicazione Android. Si esamineranno i diagrammi delle principali classi implementate nel progetto per fornire una visione completa della loro struttura.

Successivamente, si presenterà un diagramma di sequenza che illustrerà le interazioni dinamiche tra le classi principali implementate nell'app. Questo permetterà di comprendere in dettaglio le operazioni svolte dall'applicazione durante il processo di riconoscimento degli oggetti.

Inoltre, si descriveranno le animazioni implementate in Unity, che contribuiscono a rendere l'esperienza utente più coinvolgente e interattiva. Per completare il quadro della progettazione, si esplorerà anche la struttura del database utilizzato per memorizzare informazioni cruciali sulle bottiglie di vino. Questa parte è essenziale per garantire un recupero e una gestione efficiente dei dati all'interno dell'app Android.

4.1 Scelte progettuali effettuate

Nel capitolo precedente è stata fornita un'ampia panoramica degli strumenti software utilizzati, insieme alle rispettive funzionalità principali. In questa sezione, verranno descritte le versioni e le modalità di utilizzo degli strumenti. Nello specifico, questi ultimi sono:

- *Unity*: è stata selezionata la versione dell'editor *Unity 2022.3.9f1 LTS (Long-Term-Support)* al fine di garantire maggiore stabilità, rinunciando alle ultime versioni.
- *Vuforia Engine*: è stata impiegata la Versione 10.17.4.
- *PolyCam*: in questo contesto, è stata scelta la *Photo Mode* dell'applicazione. Sono state acquisite oltre 100 fotografie della bottiglia della Cantina Strappelli tramite uno smartphone al fine di creare un modello tridimensionale dell'oggetto, comprensivo di texture per il riconoscimento del colore e della grafica dell'etichetta della bottiglia di vino.
- *Blender*: poiché non sono richieste funzionalità avanzate di Blender per il progetto, è stata installata l'ultima versione disponibile, ossia la Versione 3.6.
- *Vuforia Model Target Generator*: anche in questo caso, è stata utilizzata l'ultima versione disponibile dell'applicazione, ovvero la Versione 10.17.4.
- *OpenWeather*: nel progetto, è stata utilizzata la Versione 2.5 dell'API di OpenWeather.

4.2 Diagrammi delle classi

Un *diagramma delle classi* (o *Class Diagram*, in inglese) è uno dei diagrammi più comuni utilizzati nella modellazione dei sistemi orientati agli oggetti nell’ambito dell’ingegneria del software. Questo diagramma fa parte del linguaggio di modellazione *Unified Modeling Language (UML)*, che è ampiamente utilizzato per rappresentare visivamente le strutture e le relazioni dei componenti di un sistema software.

Un diagramma delle classi prende per ciascuna classe, tre sezioni principali, ovvero:

- *Nome della classe*: è il nome della classe, che rappresenta un concetto all’interno del sistema software.
- *Attributi*: gli attributi sono le variabili che appartengono alla classe. Sono elencati sotto il nome della classe e possono includere tipi di dati e valori iniziali.
- *Metodi*: i metodi sono le funzioni o i comportamenti associati alla classe. Sono elencati sotto gli attributi e mostrano i dettagli sulla firma dei metodi, inclusi i parametri e i tipi di ritorno.

4.2.1 La classe *IotAPICaller*

La classe *IotAPICaller* è responsabile della gestione delle chiamate API per ottenere i dati provenienti dalla stazione IoT situata nel campo dell’azienda vitivinicola Strappelli. Come menzionato nel capitolo precedente, la stazione IoT carica i dati in tempo reale su una piattaforma cloud, la quale ospita un’appostia API che consente l’accesso ai dati.

L’API restituisce i dati in formato JSON, strutturati in modo simile ad una tabella di un database relazionale. Questa struttura comprende un’intestazione e un corpo di tabella, con l’aggiunta di due campi aggiuntivi, denominati *status* e *rows*.

La classe *IotAPICaller* effettua una richiesta HTTP POST con una query SQL nel corpo della richiesta per ottenere i dati più recenti caricati dalla stazione IoT sulla piattaforma cloud.

Un esempio della struttura dei dati è riportato di seguito:

```

1  {
2      "status": "succ",
3      "head": [
4          [
5              "ts",
6              "coll_time",
7              "temperature",
8              "humidity",
9              "rainfall_today",
10             "rainfall_instantaneous",
11             "rainfall_yesterday",
12             "rainfall_total",
13             "soil_temperature",
14             "soil_moisture",
15             "leaf_humidity",
16             "leaf_temperature",
17             "ext_str",
18             "ext_var1",
19             "ext_var2"
20         ],
21         "data": [
22

```

```

23      [
24          "2023-09-16 15:22:04.627",
25          "2023-09-16 15:22:04.625",
26          23.00000,
27          89.90000,
28          0.00000,
29          0.00000,
30          3.50000,
31          3.50000,
32          -3.50000,
33          -2.50000,
34          2.00000,
35          25.30000,
36          0.00000,
37          0.00000,
38          0.00000
39      ]
40  ],
41  "rows": 1
42 }

```

Listato 4.1: File JSON restituito dall'API della stazione IoT

Il primo campo denominato *"status"* mostra lo stato della richiesta HTTP inviata. Il campo *"head"*, come già accennato, rappresenta l'intestazione della tabella che descrive i dati; quest'ultima è composta dai seguenti campi inseriti all'interno di un array:

- *ts*: rappresenta un timestamp che indica quando i dati sono stati registrati.
- *coll_time*: rappresenta il momento in cui i dati sono stati effettivamente raccolti.
- *temperature*: rappresenta la temperatura in gradi Celsius.
- *humidity*: rappresenta l'umidità relativa dell'aria in percentuale.
- *rainfall_today*: rappresenta la quantità di pioggia caduta durante la giornata, espressa in millimetri.
- *rainfall_instantaneous*: rappresenta la quantità di pioggia caduta istantaneamente al momento della raccolta dei dati, espressa in millimetri.
- *rainfall_yesterday*: rappresenta la quantità di pioggia caduta nella giornata precedente, espressa in millimetri.
- *rainfall_total*: rappresenta la quantità totale di pioggia caduta fino a quel momento, espressa in millimetri.
- *soil_temperature*: rappresenta la temperatura del suolo in gradi Celsius.
- *soil_moisture*: rappresenta l'umidità del suolo, in percentuale o un'altra unità specifica.
- *leaf_humidity*: rappresenta l'umidità delle foglie delle piante, in percentuale o un'altra unità specifica.
- *leaf_temperature*: rappresenta la temperatura delle foglie delle piante in gradi Celsius.
- *ext_str, ext_var1 e ext_var2*: questi campi contengono dati aggiuntivi in formato numerico.

Il campo *data* contiene al suo interno i valori associati all'intestazione presente nel campo *head*. Infine, il campo *rows* contiene il numero di righe estratte dalla query SQL.

La classe *IotAPICaller* (Figura 4.1) ha i seguenti attributi:

- *txtLeafTemp*: temperatura della foglia nell'interfaccia grafica di Unity.
- *txtLeafHum*: umidità della foglia nell'interfaccia grafica di Unity.
- *txtTemp*: temperatura atmosferica nell'interfaccia grafica di Unity.
- *txtHum*: umidità atmosferica nell'interfaccia grafica di Unity.
- *txtRain*: millimetri di pioggia giornalieri nell'interfaccia grafica di Unity.
- *txtLatestUpd*: timestamp dei dati presentati nell'applicazione Unity.

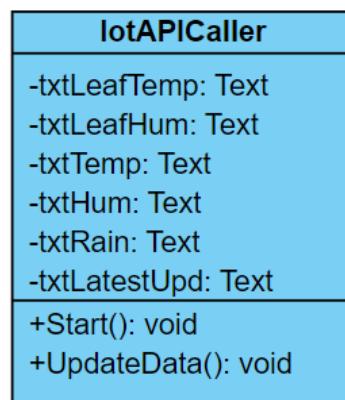


Figura 4.1: Class Diagram della classe *IotAPICaller*

Ogni variabile ha l'attributo *SerializeField* per renderla privata e visibile nell'editor di Unity in modo tale da essere vista e modificata nel valore direttamente nell'interfaccia grafica, anche se il campo è dichiarato come privato.

Infine, la classe contiene i seguenti metodi:

- *Start*: è il primo metodo eseguito al lancio dello script nella pagina "Racconta" dell'applicazione. Inizializza correttamente una richiesta HTTP impostando l'intestazione e il corpo con la query SQL per il recupero dei dati.
- *UpdateData*: aggiorna i dati visualizzati nell'applicazione eseguendo una chiamata API aggiuntiva.

4.2.2 La classe OpenWeatherAPICaller

La classe *OpenWeatherAPICaller* (Figura 4.2) si occupa di gestire i dati che provengono dall'API di OpenWeather. Quest'ultima restituisce i dati in formato JSON. Un esempio è fornito di seguito:

```

1 {
2   "coord": [
3     {
4       "lon": 13.7589,
5       "lat": 42.8142
6     },
  
```

```

7   "weather": [
8     [
9       {
10         "id": 800,
11         "main": "Clear",
12         "description": "clear sky",
13         "icon": "01d"
14       }
15     ],
16     "base": "stations",
17     "main": {
18       "temp": 299.41,
19       "feels_like": 299.41,
20       "temp_min": 295.96,
21       "temp_max": 301.19,
22       "pressure": 1016,
23       "humidity": 65,
24       "sea_level": 1016,
25       "grnd_level": 992
26     },
27     "visibility": 10000,
28     "wind": {
29       {
30         "speed": 3.41,
31         "deg": 62,
32         "gust": 2.6
33       },
34     },
35     "clouds": {
36       {
37         "all": 0
38       },
39     },
40     "dt": 1694272098,
41     "sys": {
42       {
43         "type": 2,
44         "id": 2006527,
45         "country": "IT",
46         "sunrise": 1694234268,
47         "sunset": 1694280450
48       },
49       "timezone": 7200,
50       "id": 3165549,
51       "name": "Torano Nuovo",
52       "cod": 200
53     }
54   }

```

Listato 4.2: File JSON restituito dall'API OpenWeather

Il file JSON presenta diversi campi tra cui:

- *coord*: contiene le coordinate geografiche della località; esso prende i seguenti sottocampi:
 - *lon*: rappresenta la longitudine del campo da monitorare.
 - *lat*: rappresenta la latitudine del campo da monitorare.

- *weather*: contiene informazioni sulle condizioni meteorologiche attuali; esso prende i seguenti sottocampi:
 - *id*: codice numerico che rappresenta il tipo di condizione meteorologica.
 - *main*: racchiude una descrizione categorica del meteo attuale.
 - *description*: descrizione più dettagliata della condizione meteo.
 - *icon*: rappresenta un'icona associata alle condizioni meteorologiche.
- *base*: specifica la stazione meteorologica di riferimento.
- *main*: contiene informazioni sulle condizioni meteorologiche principali; esso prende i seguenti sottocampi:
 - *temp*: rappresenta la temperatura attuale in gradi Celsius.
 - *feels_like*: è la temperatura percepita in gradi Celsius.
 - *temp_min*: rappresenta la temperatura minima prevista in gradi Celsius.
 - *temp_max*: indica la temperatura massima prevista in gradi Celsius.
 - *pressure*: rappresenta la pressione atmosferica in hPa.
 - *humidity*: rappresenta l'umidità relativa in percentuale.
 - *sea_level*: rappresenta la pressione al livello del mare in hPa.
 - *grnd_level*: rappresenta la pressione al livello del suolo in hPa.
- *visibility*: rappresenta la visibilità attuale in metri.
- *wind*: questo campo contiene informazioni sul vento; esso prende i seguenti sottocampi:
 - *speed*: rappresenta la velocità del vento in metri al secondo.
 - *deg*: rappresenta la direzione del vento in gradi.
 - *gust*: rappresenta la velocità delle raffiche di vento in metri al secondo.
- *clouds*: fornisce informazioni sulle nuvole; esso prende il seguente sottocampi:
 - *all*: rappresenta la copertura nuvolosa in percentuale.
- *dt*: è il timestamp Unix che rappresenta il momento in cui sono stati acquisiti questi dati meteorologici.
- *sys*: questo campo contiene informazioni sul sistema; esso prende i seguenti sottocampi:
 - *type*: rappresenta il tipo di sistema.
 - *id*: rappresenta l'ID del sistema.
 - *country*: rappresenta il paese associato a questa località.
 - *sunrise*: rappresenta il timestamp Unix del sorgere del sole.
 - *sunset*: rappresenta il timestamp Unix del tramonto del sole.
- *timezone*: è il fuso orario della località in secondi rispetto all'UTC.
- *id*: rappresenta l'ID univoco associato alla località.
- *name*: è il nome della località.
- *cod*: rappresenta lo stato della richiesta HTTP.

La classe *OpenWeatherAPICaller* (Figura 4.2) ha il compito di trasformare la stringa JSON proveniente dall’API di OpenWeather in una serie di variabili, ciascuna corrispondente a un campo del file JSON. Per svolgere questa operazione, essa fa uso di classi strutturate in modo da organizzare e incapsulare i dati relativi alle condizioni meteorologiche in una forma facilmente gestibile all’interno dell’applicazione. La classe principale, denominata *WeatherData*, raccoglie tutti i dati relativi alle condizioni meteorologiche, mentre le altre classi vengono impiegate per rappresentare informazioni specifiche all’interno di questa struttura dati gerarchica.

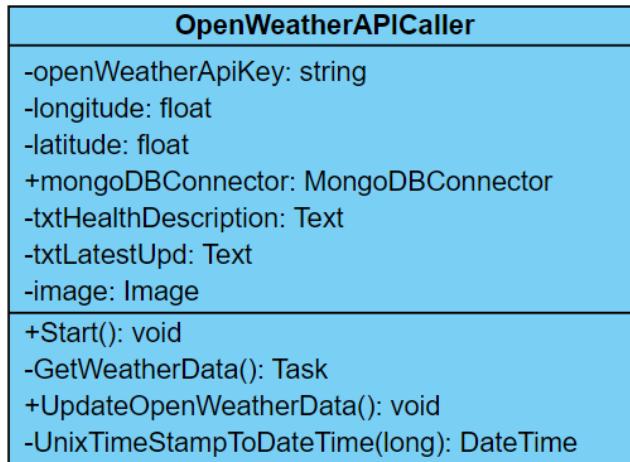


Figura 4.2: Class Diagram della classe *OpenWeatherAPICaller*

Le proprietà presenti nella classe *OpenWeatherAPICaller* sono:

- *openWeatherApiKey*: contiene la chiave che consente l’utilizzo dell’API *OpenWeather*.
- *longitude*: contiene la longitudine del campo da monitorare.
- *latitude*: contiene la latitudine del campo da monitorare.
- *mongoDBConnector*: contiene l’istanza della classe *MongoDBConnector* che si occupa di gestire la connessione con il database MongoDB (essa verrà descritta approfonditamente in seguito).
- *txtHealthDescription*: rappresenta il testo all’interno dell’interfaccia grafica di Unity relativo alla descrizione sistetica dello stato di salute del vigneto.
- *txtLatestUpd*: rappresenta il testo all’interno dell’interfaccia grafica di Unity relativo all’ultimo aggiornamento dei dati ottenuti dall’API *OpenWeather*.
- *image*: contiene il riferimento all’immagine presente nell’interfaccia grafica di Unity che mostra due tipologie di immagini in base allo stato del vigneto (in salute o in stato di allerta).

I metodi presenti nella classe, sono riportati di seguito:

- *Start()*: è il metodo che viene eseguito per prima nel lancio dello script in Unity. Esso si occupa di recuperare la longitudine e latitudine dal database del campo associato alla bottiglia di vino riconosciuta.

- *GetWeatherData()*: il metodo si occupa di effettuare la chiamata dall'API di OpenWeather per poi memorizzare i risultati nelle variabili associate a ciascun campo della stringa JSON restituita dall'API.
- *UpdateOpenWeatherData*: aggiorna i dati provenienti all'API di OpenWeather richiamando il metodo *GetWeatherData*.
- *UnixTimeStampToDate*: è un metodo che converte un timestamp Unix in una data nel formato *DateTime*.

4.2.3 La classe ScriptManager

La classe *ScriptManager* (Figura 4.3) si occupa di gestire la sincronizzazione del testo con l'audio della recensione del sommelier.

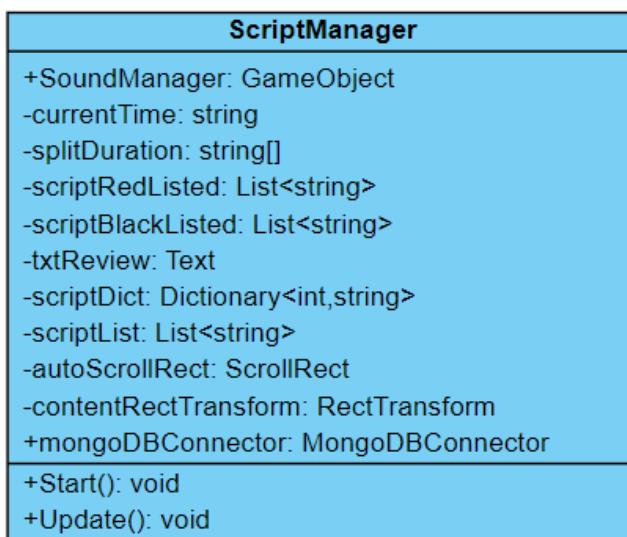


Figura 4.3: Class Diagram della classe *ScriptManager*

Gli attributi della classe sono:

- *SoundManager*: rappresenta l'oggetto che gestisce la riproduzione audio nella scena Unity.
- *currentTime*: memorizza i secondi di riproduzione attuali dell'audio in riproduzione.
- *splitDuration*: è un array di stringhe utilizzato per suddividere la variabile *currentTime* in minuti e secondi.
- *scriptRedListed*: rappresenta una lista di stringhe per memorizzare frammenti di script che devono essere visualizzati in rosso; Questi ultimi rappresentano la porzione di testo già riprodotto.
- *scriptBlackListed*: rappresenta una lista di stringhe per memorizzare frammenti di script ancora non riprodotti.
- *txtReview*: rappresenta un riferimento ad un componente di testo Unity utilizzato per visualizzare il testo associato allo script del sommelier.

- *scriptDict*: rappresenta un dizionario che memorizza coppie di valori, dove la chiave è un intero (i secondi) e il valore è una stringa (il testo dello script associato a quel secondo).
- *scriptList*: rappresenta una lista di stringhe per memorizzare l'intero script.
- *autoScrollRect*: rappresenta un riferimento ad un componente *Unity ScrollRect*. Questo attributo è serializzato in modo che possa essere assegnato nell'editor Unity.
- *contentRectTransform*: rappresenta un riferimento ad un componente *Unity RectTransform* utilizzato per gestire il layout del contenuto all'interno del *ScrollRect*.
- *mongoDBConnector*: rappresenta un oggetto della classe *MongoDBConnector* utilizzato per ottenere dati dal database MongoDB.

I metodi presenti nella classe sono i seguenti:

- *Start()*: questo metodo è chiamato all'avvio del *GameObject*. In particolare, acquisisce il testo della recensione associata alla bottiglia riconosciuta dal database MongoDB e visualizza il testo nello script nella variabile *txtRecensione*.
- *Update()*: questo metodo è chiamato ad ogni frame dell'applicazione. Calcola il tempo corrente, identifica i frammenti di script da evidenziare in base al tempo corrente di esecuzione dell'audio della recensione, aggiorna il testo in *txtRecensione*, calcola la posizione di scorrimento verticale per la visibilità del testo evidenziato e aggiorna il valore della barra di scorrimento verticale del componente *autoScrollRect* in base alla posizione del testo evidenziato.

4.2.4 La classe MongoDBConnector

La classe *MongoDBConnector* (Figura 4.4) si occupa di gestire la connessione con il database MongoDB che contiene i dati relativi al prodotto vitivinicolo identificato. La struttura del database verrà descritta in seguito.

In Figura 4.4 è mostrato il Class Diagram della classe *MongoDBConnector* di cui verranno descritti attributi e metodi presenti.

La classe presenta quattro attributi, tra cui:

- *apiKey*: contiene la chiave API che consente di accedere al database MongoDB.
- *url*: l'attributo contiene l'URL dell'API di MongoDB che consente di effettuare la ricerca nel database di un particolare record con l'operazione *findOne*.
- *jsonData*: contiene un attributo che memorizza i dati JSON da inviare nel corpo della richiesta HTTP all'API di MongoDB.
- *onDataReceived*: memorizza un attributo che contiene una funzione di callback da eseguire quando vengono ricevuti i dati dal database MongoDB.

La classe contiene un solo metodo chiamato *GetDataAsync* utilizzato per avviare un processo di recupero dei dati asincrono; tale processo accetta due parametri, ovvero:

- *callback*: un oggetto di tipo *Action* che specifica la funzione di callback da eseguire quando vengono ricevuti dati.
- *queryType*: una stringa che specifica il tipo di query da eseguire.

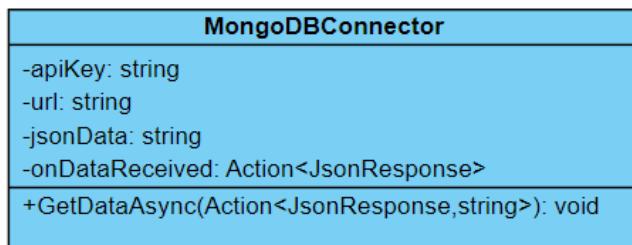


Figura 4.4: Class Diagram della classe `MongoDBConnector`

All'interno di questo metodo viene costruita una stringa di richiesta JSON in base al tipo di query richiesta; i dati pertinenti vengono ottenuti dal database MongoDB utilizzando un oggetto `UnityWebRequest`. Successivamente, la risposta viene elaborata. Se la richiesta ha successo, la risposta JSON dal database viene deserializzata in un oggetto `JsonResponse` e, quindi, passata alla funzione di callback specificata in `onDataReceived`. Nel caso in cui la richiesta non abbia successo, viene registrato un messaggio di errore e la funzione di callback viene chiamata con un parametro nullo per indicare la presenza di un errore.

4.3 Sequence Diagram

4.3.1 Introduzione

Un *diagramma di sequenza*, o *Sequence Diagram*, è uno dei diagrammi UML (*Unified Modeling Language*) utilizzati nella progettazione del software e nell'analisi dei sistemi. Questo tipo di diagramma viene utilizzato per visualizzare l'interazione tra oggetti o componenti all'interno di un sistema in un momento specifico nel tempo. In altre parole, esso mostra come gli oggetti comunicano tra loro e in che sequenza avvengono tali interazioni.

Di seguito, verranno illustrati alcuni elementi chiave che si possono trovare in un diagramma di sequenza:

- *Oggetti*: rappresentano le entità o le componenti del sistema coinvolti nell'interazione.
- *Linee di vita*: sono linee verticali che si estendono dagli oggetti e rappresentano il periodo di tempo in cui un oggetto è attivo e coinvolto nell'interazione.
- *Messaggi*: sono frecce orizzontali che collegano gli oggetti e rappresentano le comunicazioni o le chiamate di metodo tra gli oggetti. Possono essere annotate con informazioni aggiuntive, come i parametri dei metodi chiamati.
- *Attivazioni*: sono rappresentate da barre verticali sopra una linea di vita e indicano il periodo in cui un oggetto sta eseguendo una determinata operazione o un metodo.

I diagrammi di sequenza sono utili per comprendere il comportamento dinamico di un sistema e per identificare potenziali problemi o inefficienze nelle interazioni tra gli oggetti. Sono ampiamente utilizzati nella fase di progettazione e analisi dei sistemi software per documentare e comunicare le interazioni tra le parti del sistema.

4.3.2 Sequence Diagram del progetto

Nel Sequence Diagram in Figura 4.5 viene mostrato il flusso di esecuzione standard che va dalla visualizzazione del menù principale fino alla pagina dedicata alla sezione desiderata dall'utente.

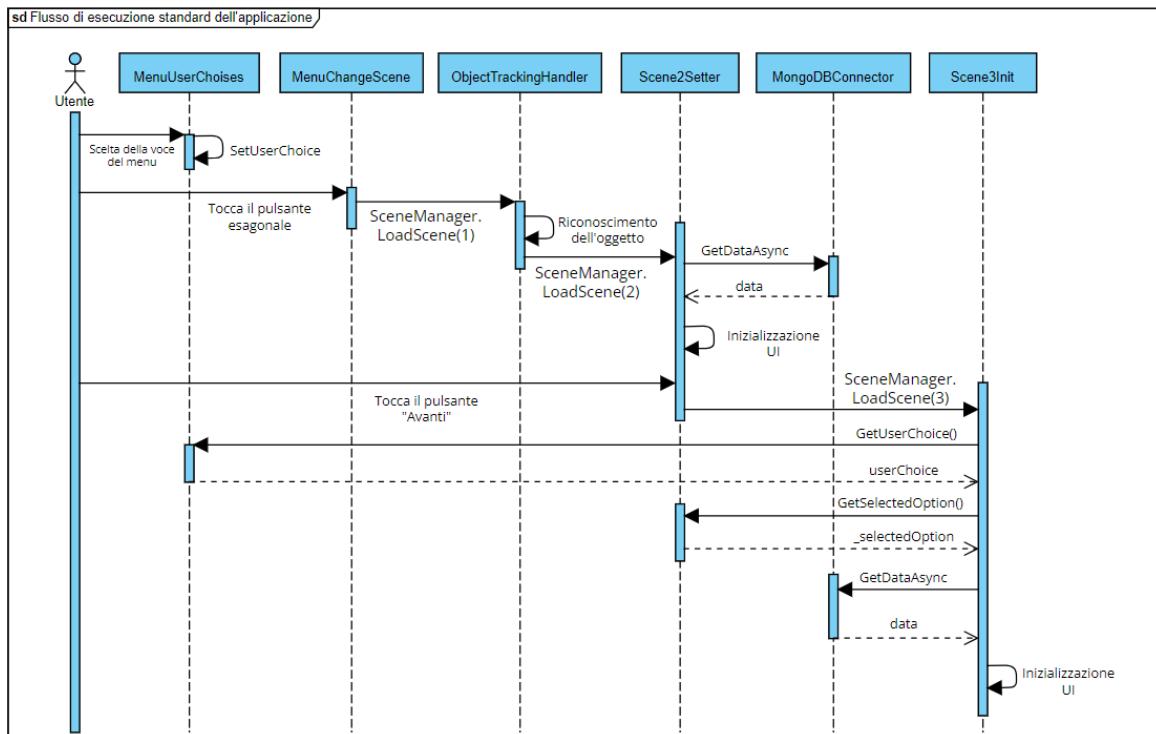


Figura 4.5: Sequence Diagram del flusso di esecuzione principale dell'applicazione

Il flusso dell'applicazione procede come di seguito specificato.

Dopo che l'animazione del menù principale ha visualizzato le tre opzioni relative alle sezioni "Osserva", "Ascolta" e "Racconta", l'utente seleziona il pulsante corrispondente alla sezione di suo interesse. La classe *MenuChangeChoices*, attraverso il metodo *SetUserChoice*, registra il nome della sezione desiderata nella variabile *userChoice*. Successivamente, l'animazione mostra il pulsante esagonale che consente di passare alla scena successiva dedicata al riconoscimento del prodotto vitivinicolo tramite il comando *SceneManager.LoadScene(1)*.

L'utente accede, quindi, alla scena dedicata al riconoscimento del prodotto vitivinicolo. Una volta che l'oggetto è stato riconosciuto, la classe *ObjectTrackingHandler* esegue il comando *SceneManager.LoadScene(2)* per caricare la scena successiva.

Nella nuova scena, vengono richiesti i dati relativi alla bottiglia di vino riconosciuta al database MongoDB. Questi dati vengono utilizzati per popolare i campi testuali presenti nell'interfaccia grafica di Unity, come il nome della cantina e la qualità del vino.

Una volta che la scena è stata inizializzata correttamente con i dati relativi al prodotto vitivinicolo, l'utente seleziona l'annata desiderata e preme il pulsante "Avanti", il che avvia il caricamento dell'ultima scena dell'applicazione chiamata *SceneManager.LoadScene(3)*.

All'avvio dell'ultima scena dell'applicazione, questa richiede alla classe *MenuUserChoices* tramite il metodo pubblico *GetUserChoice*, la stringa contenente la scelta effettuata dall'utente nel menu principale, precedentemente memorizzata nella variabile *userChoice*. Successivamente, la classe *Scene3Init* acquisisce l'annata scelta dall'utente tramite il metodo *GetSelectedOption* e utilizza questi dati per popolare gli elementi dell'interfaccia grafica, richiedendo ulteriori informazioni alla classe *MongoDBConnector*.

4.4 Animazioni in Unity

4.4.1 Introduzione

Le animazioni in Unity sono un elemento fondamentale per creare giochi interattivi e applicazioni. Le animazioni consentono di far muovere gli oggetti, i personaggi, le telecamere e molto altro all'interno del progetto Unity.

Di seguito, è riportata una panoramica del funzionamento delle animazioni in Unity:

- *Animator Controller*: è il componente principale per gestire le animazioni in Unity. Questo controller definisce gli stati, le transizioni e le regole per la riproduzione delle animazioni.
- *Animation Clips*: Un Animation Clip è un file che contiene un'animazione specifica. Puoi crearli importando file da programmi di modellazione 3D o crearli direttamente in Unity utilizzando l'Editor di animazione. Questi clip vengono poi collegati all'Animator Controller.
- *Animator Window*: è un'interfaccia utente visuale che consente di creare e gestire le transizioni tra gli stati dell'Animator Controller. È possibile aggiungere transizioni tra diversi stati e definire le condizioni che le attivano.
- *Layers*: gli Animator Controller possono avere più layer, che consentono di gestire animazioni sovrapposte o di priorità diverse.
- *Blend Trees*: consentono di miscelare più animazioni in base a valori specifici, come direzione e velocità. Questi sono utili per controllare animazioni complesse.
- *Parametri*: consentono di utilizzare parametri per controllare le transizioni tra gli stati.
- *Scripting*: è possibile controllare l'animazione tramite script C# in Unity.
- *Importazione di animazioni*: è possibile importare animazioni create in programmi come Blender e collegarle all'Animator Controller.
- *Esecuzione in tempo reale*: le animazioni in Unity possono essere eseguite in tempo reale, consentendo interazioni dinamiche con il mondo di gioco.

4.4.2 Le animazioni utilizzate nel progetto

Menù principale

In Figura 4.6, è riportata la struttura dell'animator controller denominato *Menu Controller*. Nel momento in cui l'applicazione Android viene eseguita, l'animator che si trova nello stato *Entry*, attiva automaticamente l'animazione *MenuFadeIn* che permette di visualizzare le tre voci del menù principale descritte nei capitoli precedenti con un effetto *FadeIn* in ingresso.

Successivamente, l'applicazione rimane in attesa di un input da parte dell'utente. Quest'ultimo, nel momento in cui effettua la scelta, attiva l'animazione associata alle tre voci presenti nel menù, rispettivamente *ChoiceOsserva*, *ChoiceAscolta* e *ChoiceRacconta*. Ciascuna di queste animazioni fa passare automaticamente allo stato *Scene1_Idle*.

L'applicazione mostrerà un pulsante a forma di esagono che attende l'input dell'utente. Nel momento in cui il pulsante verrà premuto, si attiverà la scena dedicata al riconoscimento del prodotto vitivinicolo.

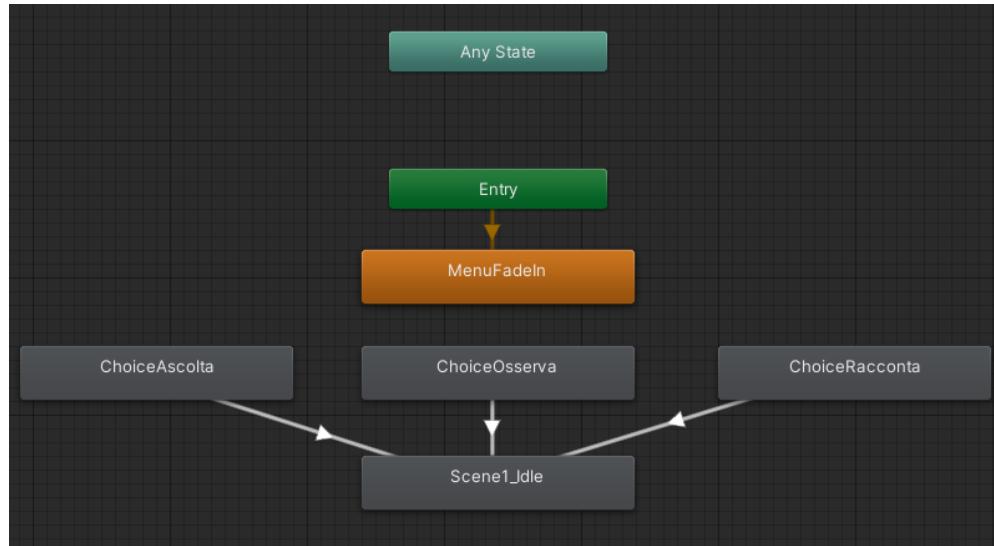


Figura 4.6: La struttura dell'Animator controller *Menu Controller*

Scelta dell'annata

Come già anticipato nei capitoli precedenti, una volta che il prodotto vitivinicolo è stato riconosciuto, l'applicazione carica la scena dedicata alla selezione dell'annata del vino. L'Animator controller chiamato "*Scene2IntroController*" (Figura 4.7) all'avvio della nuova scena, è nello stato *Entry*, che attiva, tramite una transizione, l'animazione *Scene2Intro*. Quest'ultima si occupa di effettuare lo scorrimento della finestra dal basso verso l'alto fino a metà schermo per permettere all'utente di scegliere l'annata della bottiglia di vino di suo interesse.

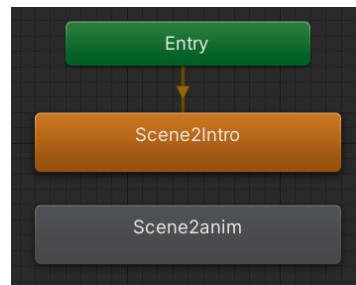


Figura 4.7: La struttura dell'Animator controller "Scene2IntroController"

Dopo aver scelto l'annata, l'utente preme il pulsante "Avanti" che attiva la scena *Scene2anim*; quest'ultima effettua lo scorrimento completo della finestra per creare un effetto di transizione verso l'ultima scena dell'applicazione che mostra le tre sezioni "Osserva", "Ascolta" e "Racconta".

4.5 Il database MongoDB

Il database MongoDB, come già accennato, contiene i dati relativi ai vigneti monitorati dall'azienda Trace Technologies. Ogni documento MongoDB contiene i dati di un vigneto strutturati come in Figura 4.8:

Più specificatamente, i campi presenti in ogni documento MongoDB sono i seguenti:

- *Nome*: rappresenta il nome univoco del prodotto vitivinicolo.

```

1  _id: ObjectId('6501b3877f9b502fb7ec06c6')
2  Nome: "strappelli_montepulciano"
3  Titolo1: "Cantina Strappelli"
4  Titolo2: "Colline Teramane Montepulciano d'Abruzzo DOCG"
5  Latitudine: 42.814172
6  Longitudine: 13.758886
7  ▶ Descrizione: Object
8  LuogoProd: "Torano Nuovo (TE)"
9  ▶ ScriptSommelier: Object

```

Figura 4.8: Documento MongoDB della cantina Strappelli

- *Titolo1*: contiene il nome che appare nelle intestazioni del prodotto vitivinicolo nelle varie sezioni dell'applicazione Android.
- *Titolo2*: contiene le sotto-intestazioni del prodotto vitivinicolo nelle varie sezioni dell'applicazione Android.
- *Latitudine*: memorizza la latitudine del campo associato al vigneto.
- *Longitudine*: memorizza la longitudine del campo associato al vigneto.
- *Descrizione*: contiene una breve descrizione delle caratteristiche del vino associato alla bottiglia riconosciuta suddivisa per annata.
- *LuogoProd*: contiene il luogo in cui il vino viene prodotto.
- *ScriptSommelier*: memorizza un dizionario per ogni annata, in cui ad ogni timestamp in secondi è associata una stringa di testo. Questo campo viene utilizzato nella sezione "Ascolta" descritta in precedenza per ottenere il testo sincronizzato con l'audio (Figura 4.9).

```

    ▶ ScriptSommelier: Object
      ▶ 2018: Object
        0: "Il Montepulciano d'Abruzzo della"
        1: "Cantina Strappelli è un vino rosso di"
        3: "eccellente qualità che incarna"
        5: "l'autenticità e la passione della"
        7: "regione dell'Abruzzo in Italia."
        9: "Questo vino è ottenuto con cura da uve"
        11: "Montepulciano selezionate a mano"
        13: "provenienti dai vigneti di famiglia,"
        15: "coltivati su terreni ricchi di storia e"
        17: "tradizione vitivinicola."
        20: "Nel bicchiere, il Montepulciano d'Abruzzo"
        22: "della Cantina Strappelli si presenta con"
        25: "un colore rubino profondo e intenso, con"
        27: "riflessi violacei che ne evidenziano la"
        29: "giovinezza. Al naso, offre un bouquet"

```

Figura 4.9: Struttura del campo *ScriptSommelier*

CAPITOLO 5

Implementazione e manuale utente

Questo capitolo assume una notevole importanza poiché tratterà diversi aspetti cruciali del progetto. Si inizierà esaminando la procedura per la creazione e l'importazione di nuovi oggetti tridimensionali all'interno dell'applicazione Unity.

Successivamente, verrà affrontata l'implementazione in codice C# delle classi descritte nel capitolo precedente. Inoltre, verrà fornita una descrizione dettagliata del processo di creazione delle animazioni utilizzate nel menù principale e nella scena dedicata alla selezione dell'annata del prodotto vitivinicolo.

Infine, nell'ultima sezione, verrà presentato un manuale utente completo che guiderà gli utenti nell'utilizzo dell'applicazione in modo semplice ed efficace.

5.1 Creazione ed importazione di un modello tridimensionale di un prodotto

5.1.1 Creazione del modello tridimensionale dell'oggetto tramite smartphone

In questa sezione verrà descritto il processo di creazione ed importazione di un modello tridimensionale di un prodotto vitivinicolo per consentire successivamente a Vuforia Engine di riconoscerlo tramite l'applicazione Android sviluppata in Unity.

Come precedentemente menzionato, il primo passo consiste nella generazione di un modello tridimensionale utilizzando l'applicazione mobile *PolyCam*. Dopo aver creato un account gratuito tramite l'applicazione, è stata scelta la modalità *Photo Mode* per scattare una serie di fotografie dell'oggetto da utilizzare per la costruzione del modello tridimensionale. Nel caso del progetto, per ottenere un risultato ottimale, sono state scattate oltre 100 fotografie della bottiglia di vino ed è stato necessario il posizionamento dell'oggetto all'interno di un set fotografico ben preparato, in quanto è importante eliminare qualsiasi "rumore" o "disturbo" presente nello sfondo. Infatti, è consigliato posizionare un pannello o un cartoncino scuro dietro all'oggetto per evitare che altri oggetti interferiscano con la costruzione del modello tridimensionale. Nel progetto, è stata utilizzata una scatola di cartone posta dietro alla bottiglia di vino della cantina Strappelli come mostrato in Figura 5.1.

Inoltre, le foto devono essere scattate in modo tale da ottenere una panoramica completa a 360° della bottiglia. Per raggiungere questo obiettivo, essa è stata posizionata su una piattaforma girevole, consentendone, così, la rotazione durante l'acquisizione delle immagini in sequenza.



Figura 5.1: Il set utilizzato per creare il modello 3D

Una volta ottenute le immagini dell’oggetto, esse vengono inviate al server di PolyCam. In poco tempo, viene fornito un modello tridimensionale dell’oggetto desiderato nel formato glTF.

5.1.2 Raffinamento del modello tridimensionale con Blender

Il modello tridimensionale fornito da *PolyCam* solitamente mostra delle imperfezioni dovute al rumore presente nelle immagini acquisite. Nel caso della bottiglia di vino, è stato necessario rimuovere, tramite l’utilizzo di Blender, alcune imperfezioni che si trovavano nella parte superiore e alla base dell’oggetto, poiché rappresentavano porzioni dello sfondo erroneamente incluse nel modello tridimensionale. Di seguito sono riportati il modello prima (Figura 5.2) e dopo (Figura 5.3) le correzioni effettuate in Blender.

5.1.3 Creazione ed importazione di un modello in Vuforia Engine

Una volta corretto il modello con Blender, è possibile esportarlo nel formato glTF richiesto da Vuforia. Successivamente, è necessario creare un account e una licenza gratuita sul portale degli sviluppatori di Vuforia.

Una volta ottenuta la licenza, è necessario importare il file glTF precedentemente creato all’interno di *Vuforia Model Target Generator*. Questo strumento rielaborerà il modello in modo tale da essere poi importato tramite il *Package Manager* di Unity. Successivamente, si procede ad importare *Vuforia Engine* tramite l’*Asset Store* di Unity. Nella scena dedicata al riconoscimento delle bottiglie, è stato inserito un oggetto chiamato *Model Target* che contiene gli strumenti del motore grafico Vuforia utili per eseguire il riconoscimento del prodotto vitivinicolo tra cui il modello tridimensionale creato in precedenza. L’oggetto *Model Target* è mostrato in Figura 5.4.

All’interno di quest’ultimo, si trova uno script C# chiamato *ObjectTrackingHandler* che include una classe omonima e un metodo specifico denominato *OnTrackingFound*. Quest’ultimo consente, per prima cosa, di salvare il nome dell’oggetto riconosciuto in una variabile



Figura 5.2: Modello 3D prima dell'utilizzo di Blender

chiamata *trackedObjectName* per poi, in secondo luogo, caricare la scena successiva dedicata alla selezione dell'annata di riferimento del vino.

5.2 Implementazione delle classi principali

Nel capitolo precedente, sono stati forniti i Class Diagram delle classi più importanti che permettono di implementare le funzionalità principali dell'applicazione Android. In questa sezione, si mostrerà il codice per ognuna delle classi create.

5.2.1 Implementazione della classe *IotAPICaller*

Come già accennato in precedenza, la classe *IotAPICaller* permette all'applicazione Android di ottenere i dati in tempo reale dalla stazione IoT presente nel vigneto di proprietà della cantina Strappelli. Nel Listato 5.1 è mostrato il codice della classe.

```

1  using Newtonsoft.Json;
2  using System;
3  using System.Collections.Generic;
4  using System.Globalization;
5  using System.IO;
6  using System.Net;
7  using System.Text;
8  using System.Threading.Tasks;
9  using UnityEngine;
10 using UnityEngine.Networking;
11 using UnityEngine.UI;
12
13
14
15 public class IotAPICaller: MonoBehaviour
16 {
17     public string status { get; set; }
18     public List<string> head { get; set; }
19     public List<List<string>> data { get; set; }
20     public int rows { get; set; }
21
22 }
```

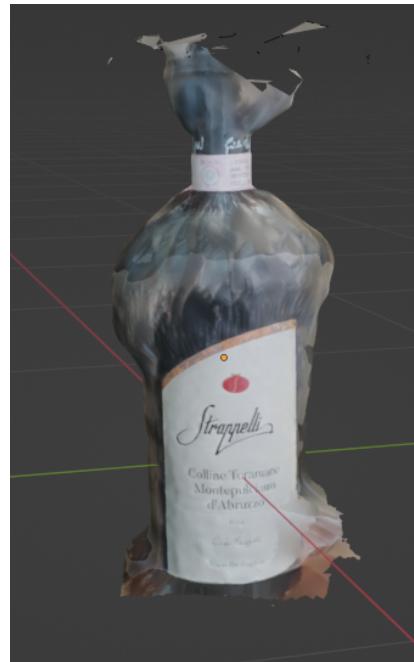


Figura 5.3: Modello 3D dopo l'utilizzo di Blender

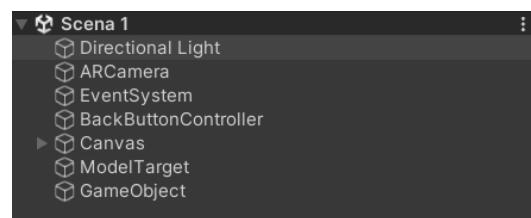


Figura 5.4: Posizionamento dell'oggetto *ModelTarget* nella scena

```

23
24 [SerializeField] private Text txtLeafTemp;
25 [SerializeField] private Text txtLeafHum;
26 [SerializeField] private Text txtTemp;
27 [SerializeField] private Text txtHum;
28 [SerializeField] private Text txtRain;
29 [SerializeField] private Text txtLatestUpd;
30 void Start()
31 {
32     HttpWebRequest request = (HttpWebRequest)WebRequest.Create("http://13.51.174.103:6041/rest/sql");
33     request.ContentType = "application/json";
34     request.Method = "POST";
35     request.Headers.Add("Authorization", "Basic " + "cm9vdDp0YW9zZGF0YQ==");
36     request.ContentType = "application/json";
37
38     Stream dataStream = request.GetRequestStream();
39     string sql = "select * from station1.sensor_data where ts >= NOW - 48h order by ts DESC limit 1";
40     byte[] byteArray = Encoding.ASCII.GetBytes(sql);
41     dataStream.Write(byteArray, 0, byteArray.Length);
42     HttpWebResponse response = (HttpWebResponse)request.GetResponse();
43     string responseString = new StreamReader(response.GetResponseStream()).ReadToEnd();
44
45     IoTAPICaller apiData = JsonConvert.DeserializeObject<IoTAPICaller>(responseString);
46
47     txtLeafTemp.text = (apiData.data[0][11].ToString().Substring(0, 5) + " \u00b0C");
48     txtLeafHum.text = (apiData.data[0][10].ToString().Substring(0, 3) + "%");
49     txtTemp.text = (apiData.data[0][2].ToString().Substring(0, 5) + " \u00b0C");
50     txtHum.text = (apiData.data[0][3].ToString().Substring(0, 5) + "%");
51     txtRain.text = (apiData.data[0][4].ToString().Substring(0, 3) + " mm");

```

```

52     DateTime latestUpdDate = DateTime.ParseExact(apiData.data[0][1].ToString().Substring(0, 10), "yyyy-MM-dd",
53         CultureInfo.InvariantCulture);
54     DateTime latestUpdTime = DateTime.ParseExact(apiData.data[0][1].ToString().Substring(11, 5), "HH:mm",
55         CultureInfo.InvariantCulture);
56     txtLatestUpd.text = (latestUpdDate.ToString("dd-MM-yyyy") + " " + latestUpdTime.AddHours(2).ToString("HH:
57         mm")));
58
59
60     public void UpdateData()
61     {
62         txtLeafTemp.text = "";
63         txtLeafHum.text = "";
64         txtTemp.text = "";
65         txtHum.text = "";
66         txtRain.text = "";
67         txtLatestUpd.text = "";
68
69         Start();
70     }
71 }
72 }
```

Listato 5.1: Codice sorgente dello script *IoTAPICaller*

L'inizio dello script include una serie di dichiarazioni di namespace, che forniscono accesso a librerie di terze parti e funzionalità di base. Alcuni di questi namespace includono *Newtonsoft.Json* per il parsing JSON, *System* namespace principale di C#, *System.Collections.Generic* per liste generiche, *System.Globalization* per il supporto multilingue, *System.IO* per operazioni di I/O, *System.Net* per comunicazione di rete, *System.Text* per lavorare con stringhe e *UnityEngine* per le funzionalità specifiche di Unity.

La classe *IoTAPICaller* è definita come una classe derivata da *MonoBehaviour*, il che significa che è progettata per essere associata ad un oggetto *GameObject* in Unity. Questa classe ha diverse proprietà pubbliche, come *status*, *head*, *data*, *rows*, che rappresentano i dati ricevuti dall'API come già anticipato nel capitolo precedente.

La classe contiene campi serializzati (*[SerializeField]*) che rappresentano oggetti di testo (di tipo *Text*) nell'interfaccia utente Unity. Questi campi verranno popolati con dati ricevuti dall'API e utilizzati per visualizzare le informazioni sull'interfaccia utente.

Il metodo *Start()* è un metodo speciale di Unity che viene chiamato quando il componente viene inizializzato. All'interno di questo metodo:

- Viene creata una richiesta HTTP di tipo POST (*HttpWebRequest*) per l'URL specificato (*http://13.51.174.103:6041/rest/sql*).
- Vengono impostati vari dettagli sulla richiesta, come il tipo di contenuto (*application/json*), il metodo (*POST*), e l'autorizzazione (tramite un'intestazione *Authorization*) per poter accedere ai dati presenti nella piattaforma cloud della stazione IoT.
- Viene creato un flusso di dati (*dataStream*) per l'invio di essi nella richiesta, e viene inviata una query SQL al server IoT. Quest'ultima richiede tutti i campi della tabella *station1.sensor_data* in una finestra temporale di 48 ore prendendo soltanto il primo record che corrisponde al dato più recente grazie al comando di ordinamento decrescente eseguito sul campo timestamp *ts*.
- Viene ricevuta una risposta HTTP dal server IoT e il suo contenuto viene letto e memorizzato come una stringa.

- La stringa di risposta JSON viene deserializzata utilizzando *JsonConvert* in un oggetto *IotAPICaller*. I dati ricevuti vengono estratti e utilizzati per aggiornare i campi di testo nell’interfaccia utente Unity.
- Il metodo *UpdateData()* cancella il contenuto dei campi di testo e, quindi, richiama il metodo *Start()* per effettuare una nuova richiesta e aggiornare i dati.

5.2.2 Implementazione della classe OpenWeatherAPICaller

La seconda classe implementata nel progetto, chiamata *OpenWeatherAPICaller*, si occupa di richiamare i dati provenienti dall’API di OpenWeather, come già descritto nel capitolo precedente. Nel Listato 5.2 è presente l’implementazione in C# della classe.

```
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using System.Threading.Tasks;
5  using UnityEngine;
6  using UnityEngine.Networking;
7  using UnityEngine.UI;
8
9
10 // Define a class structure to match the JSON data
11 [System.Serializable]
12 public class Coord
13 {
14     public float lon;
15     public float lat;
16 }
17
18 [System.Serializable]
19 public class Weather
20 {
21     public int id;
22     public string main;
23     public string description;
24     public string icon;
25 }
26
27 [System.Serializable]
28 public class Main
29 {
30     public float temp;
31     public float feels_like;
32     public float temp_min;
33     public float temp_max;
34     public int pressure;
35     public int humidity;
36     public int sea_level;
37     public int grnd_level;
38 }
39
40 [System.Serializable]
41 public class Wind
42 {
43     public float speed;
44     public int deg;
45     public float gust;
46 }
47
48 [System.Serializable]
49 public class Clouds
50 {
51     public int all;
52 }
53
54 [System.Serializable]
55 public class Sys
```

```
56 {
57     public int type;
58     public int id;
59     public string country;
60     public int sunrise;
61     public int sunset;
62 }
63
64 [System.Serializable]
65 public class WeatherData
66 {
67     public Coord coord;
68     public Weather[] weather;
69     public string baseData;
70     public Main main;
71     public int visibility;
72     public Wind wind;
73     public Clouds clouds;
74     public int dt;
75     public Sys sys;
76     public int timezone;
77     public int id;
78     public string name;
79     public int cod;
80 }
81
82 public class OpenWeatherAPICaller : MonoBehaviour
83 {
84     private string openWeatherApiKey = "ea5f560544db2e6226f8c5da0c559f61";
85
86     private float longitude = 0f;
87     private float latitude = 0f;
88
89     public MongoDBConnector mongoDBConnector;
90
91     [SerializeField] private Image image;
92     [SerializeField] private Text txtHealthDescription;
93     [SerializeField] private Text txtLatestUpd;
94
95     // Start is called before the first frame update
96     async void Start()
97     {
98         if (longitude == 0f || latitude == 0f)
99         {
100             mongoDBConnector.GetDataAsync((data) =>
101             {
102                 if (data != null)
103                 {
104                     latitude = data.document.Latitudine;
105                     longitude = data.document.Longitudine;
106                 }
107             }, "lat_long");
108         }
109
110         await GetWeaherData();
111     }
112
113     private async Task GetWeaherData()
114     {
115         string url = $"https://api.openweathermap.org/data/2.5/weather?lat={latitude}&lon={longitude}&appid={openWeatherApiKey}&units=metric";
116
117         using (UnityWebRequest webRequest = UnityWebRequest.Get(url))
118         {
119             var asyncOperation = webRequest.SendWebRequest();
120
121             while (!asyncOperation.isDone)
122             {
123                 await Task.Yield();
124             }
125
126             if (webRequest.result == UnityWebRequest.Result.ConnectionError || webRequest.result ==
```

```
127     UnityWebRequest.Result.ProtocolError)
128     {
129         Debug.LogError("Errore nella richiesta: " + webRequest.error);
130         txtLatestUpd.text = "";
131     }
132     else
133     {
134         string json = webRequest.downloadHandler.text;
135
136         WeatherData openWeatherData = JsonUtility.FromJson<WeatherData>(json);
137
138         DateTime dateTime = UnixTimeStampToDate((openWeatherData.dt));
139
140         string formattedDate = dateTime.ToString("dd/MM/yyyy HH:mm:ss", new System.Globalization.CultureInfo("it-IT"));
141
142         txtLatestUpd.text = formattedDate;
143
144         // temporale
145
146         if ((openWeatherData.weather[0].id > 201 && openWeatherData.weather[0].id < 233))
147         {
148             image.sprite = Resources.Load<Sprite>("Images/attention");
149             txtHealthDescription.text = "Oggi è una giornata difficile per il vigneto. C'è un temporale in corso";
150             if (openWeatherData.wind.speed > 6f)
151             {
152                 txtHealthDescription.text = "Oggi è una giornata difficile per il vigneto. C'è un temporale in corso con forte vento";
153             }
154         else if ((openWeatherData.weather[0].id > 310 && openWeatherData.weather[0].id < 322) ||
155                  (openWeatherData.weather[0].id > 502 && openWeatherData.weather[0].id < 533))
156         {
157             image.sprite = Resources.Load<Sprite>("Images/attention");
158             txtHealthDescription.text = "Oggi nel vigneto sta piovendo molto... Potrebbe aumentare il rischio di malattie nel vigneto";
159             if (openWeatherData.wind.speed > 6f)
160             {
161                 txtHealthDescription.text = "Oggi è una giornata difficile per il vigneto... Sta piovendo molto e tira un forte vento. Potrebbe aumentare il rischio di malattie nel vigneto";
162             }
163         else if ((openWeatherData.weather[0].id >= 600 && openWeatherData.weather[0].id < 623))
164         {
165             image.sprite = Resources.Load<Sprite>("Images/attention");
166             txtHealthDescription.text = "Oggi nel vigneto sta nevicando! Potrebbe aumentare il rischio di malattie nel vigneto";
167             if (openWeatherData.wind.speed > 6f)
168             {
169                 txtHealthDescription.text = "Oggi nel vigneto sta nevicando e tira un forte vento! Potrebbe aumentare il rischio di malattie nel vigneto";
170             }
171         else if ((openWeatherData.weather[0].id > 800 && openWeatherData.weather[0].id < 805))
172         {
173             image.sprite = Resources.Load<Sprite>("Images/ok");
174             txtHealthDescription.text = "Oggi va tutto bene, ci sono soltanto delle nuvole...";
175             if (openWeatherData.wind.speed > 6f)
176             {
177                 txtHealthDescription.text = "Oggi nel vigneto tira un forte vento! Potrebbe danneggiare il vigneto ...";
178             }
179         else if (openWeatherData.weather[0].id == 800)
180         {
181             image.sprite = Resources.Load<Sprite>("Images/ok");
182             txtHealthDescription.text = "Oggi va tutto bene nel vigneto!";
183         }
184     else
185     {
186         image.sprite = Resources.Load<Sprite>("Images/ok");
187         txtHealthDescription.text = "Oggi va tutto bene nel vigneto!";
188     }
189
190 }
```

```

191     }
192   }
193 }
194 }
195
196 public async void UpdateOpenWeatherData()
197 {
198   image.sprite = Resources.Load<Sprite>("Images/loading");
199
200   txtHealthDescription.text = "";
201   txtLatestUpd.text = "";
202
203   await GetWeaherData();
204 }
205
206 // Function to convert Unix timestamp to DateTime
207 private DateTime UnixTimeStampToDate(long unixTimestamp)
208 {
209   DateTime dateTime = new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
210   dateTime = dateTime.AddSeconds(unixTimestamp).ToLocalTime();
211   return dateTime;
212 }
213 }
```

Listato 5.2: Codice sorgente dello script *OpenWeatherAPICaller*

Di seguito verrà descritto il codice dettagliatamente:

- La prima parte si occupa dell'importazione delle librerie:
 - *System*, *System.Collections*, e *System.Collections.Generic*, che rappresentano gli spazi dei nomi standard di C#.
 - *System.Threading.Tasks*, utilizzata per gestire operazioni asincrone.
 - *UnityEngine*, che rappresenta il namespace di Unity.
 - *UnityEngine.Networking*, utilizzata per eseguire richieste web.
 - *UnityEngine.UI*, utilizzata per manipolare gli elementi dell'interfaccia utente nell'ambiente Unity.
- La seconda parte del codice è dedicata alla definizione di classi utili per la deserializzazione dei dati provenienti dall'API *OpenWeather*; le classi *Coord*, *Weather*, *Main*, *Wind*, *Clouds*, *Sys*, e *WeatherData* sono utilizzate per corrispondere ai dati JSON restituiti dall'API di OpenWeatherMap. Queste classi sono annotate con *[System.Serializable]* per consentire la serializzazione/deserializzazione dei dati JSON.
- Successivamente, il codice descrive le variabili di classe tra cui:
 - *openWeatherApiKey*, che contiene la chiave API per *OpenWeather*.
 - *longitude* e *latitude*, che contengono le coordinate iniziali, inizializzate a 0.
 - *mongoDBConnector*, che contiene un riferimento a un'istanza di un'altra classe per la connessione al database MongoDB.
 - *image*, *txtHealthDescription*, e *txtLatestUpd*, che rappresentano riferimenti agli oggetti UI nell'interfaccia Unity.
- Il metodo *Start()* viene chiamato all'avvio dello script Unity e verifica se *longitude* e *latitude* sono uguali a 0. Se lo sono, cerca di ottenere queste coordinate dal database MongoDB utilizzando *mongoDBConnector.GetDataAsync()*. Quindi, richiama il metodo *GetWeatherData()* in modo asincrono.
- Il metodo *GetWeatherData()*:

- Costruisce l'URL per la richiesta API *OpenWeather* utilizzando *latitude*, *longitude* e *openWeatherApiKey*, che rappresenta la chiave di licenza.
- Invia una richiesta web asincrona utilizzando *UnityWebRequest*.
- Attende il completamento della richiesta con un ciclo while.
- Se la richiesta è stata eseguita con successo, converte il risultato JSON in un oggetto della classe *WeatherData*.
- Estrae i dati desiderati dall'oggetto *WeatherData* e aggiorna l'interfaccia utente. Per individuare lo stato di salute del vigneto, si utilizzano il codice relativo alle condizioni meteo utilizzato da OpenWeather e la velocità del vento.
- Il metodo *UpdateOpenWeatherData()*:
 - viene chiamato quando si desidera aggiornare i dati meteorologici.
 - mostra un'immagine di caricamento e cancella il testo nell'interfaccia utente.
 - chiama il metodo *GetWeatherData()* in modo asincrono per aggiornare i dati presenti nella UI di Unity.
- Il metodo *UnixTimeStampToDate()* converte un timestamp UNIX in un oggetto *DateTime* localizzato.

5.2.3 Implementazione della classe ScriptManager

L'obiettivo principale di questa classe è quello di visualizzare, scorrere ed evidenziare il testo dello script del sommelier in base al tempo corrente di esecuzione della traccia audio della recensione del vino. Nel Listato 5.3 è mostrato il codice della classe.

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using System;
5  using System.Linq;
6  using System.IO;
7  using UnityEngine.UI;
8  using System.Globalization;
9  using UnityEngine.Events;
10 using UnityEngine.EventSystems;
11
12 public class ScriptManager : MonoBehaviour
13 {
14     public GameObject SoundManager;
15     private string currentTime = "";
16     private string[] splitDuration;
17     List<string> scriptRedListed = new List<string>();
18     List<string> scriptBlackListed = new List<string>();
19
20     private Dictionary<int, string> scriptDict = new Dictionary<int, string>();
21     private List<String> scriptList = new List<string>();
22
23     [SerializeField] ScrollRect autoScrollRect;
24     [SerializeField] private RectTransform contentRectTransform;
25     [SerializeField] Text txtReview;
26     public MongoDBConnector mongoDBConnector;
27
28     // Start is called before the first frame update
29     void Start()
30     {
31         mongoDBConnector.GetDataAsync((data) =>
32         {
33             if (data != null)
34             {

```

```
36     Dictionary<string, string> scriptSommelierSelected = data.document.ScriptSommelier[Scene2Setter.  
37         GetSelectedOption()];  
38  
39     foreach (var kvp in scriptSommelierSelected)  
40     {  
41         scriptDict[int.Parse(kvp.Key)] = kvp.Value; // Add the entry to the dictionary  
42     }  
43  
44     }  
45     else  
46     {  
47         Debug.LogError("Impossibile connettersi al DB!");  
48     }  
49  
50 }, "script");  
51  
52     foreach (KeyValuePair<int, string> scriptScorePair in scriptDict)  
53     {  
54         scriptList.Add(scriptScorePair.Value);  
55     }  
56  
57     txtReview.text = (string.Join("\n", scriptList));  
58  
59     if (autoScrollRect == null) autoScrollRect = GetComponent<ScrollRect>();  
60  
61 }  
62  
63  
64  
65 // Update is called once per frame  
66 void Update()  
67 {  
68     currentTime = SoundManager.GetComponent<SoundManager>().getCurrentTime();  
69  
70     int script_minutes = 0;  
71     int script_seconds = 0;  
72  
73     splitDuration = currentTime.Split(':');  
74  
75     if (splitDuration.Length > 1)  
76     {  
77         if (!(String.IsNullOrEmpty(splitDuration[0])) || !(String.IsNullOrEmpty(splitDuration[1])))  
78         {  
79             script_minutes = Int32.Parse(splitDuration[0]);  
80             script_seconds = Int32.Parse(splitDuration[1]);  
81         }  
82     }  
83  
84     int totalSeconds = 0;  
85  
86     totalSeconds = script_minutes * 60 + script_seconds;  
87  
88     scriptRedListed = scriptDict  
89         .Where(item => item.Key <= totalSeconds)  
90         .Select(item => item.Value)  
91         .ToList();  
92  
93     scriptBlackListed = scriptDict  
94         .Where(item => item.Key > totalSeconds)  
95         .Select(item => item.Value)  
96         .ToList();  
97  
98     if (scriptRedListed.Count > 0)  
99     {  
100         scriptBlackListed.Insert(0, "");  
101     }  
102  
103     txtReview.text = "<color=#8F1338>" + string.Join("\n", scriptRedListed) + "</color>" +  
104         string.Join("\n", scriptBlackListed);  
105  
106 }
```

```

107 // Calculate the total height of the content
108 float totalContentHeight = contentRectTransform.rect.height;
109
110 // Calculate the height of a single line of text in the ScrollView
111 float lineHeight = txtReview.fontSize + txtReview.lineSpacing;
112
113 float highlightedTextPosition = (scriptRedListed.Count * lineHeight);
114
115 // Calculate the height of the ScrollView viewport
116 float scrollViewportHeight = autoScrollRect.viewport.rect.height;
117
118 // Calculate the maximum vertical scroll position (bottom-most position of the ScrollView)
119 float maxVerticalScrollPosition = totalContentHeight - scrollViewportHeight;
120
121 // Calculate the target vertical scroll position based on the highlighted text position
122 float targetVerticalScrollPosition = Mathf.Clamp(highlightedTextPosition, 0f, maxVerticalScrollPosition);
123
124 // Calculate the normalized vertical scroll position (between 0 and 1)
125 float normalizedScrollPosition = targetVerticalScrollPosition / maxVerticalScrollPosition;
126
127 if (normalizedScrollPosition < 0.058f)
128 {
129     autoScrollRect.verticalScrollbar.value = 1f;
130 }
131 }else if(normalizedScrollPosition > 0.058f && normalizedScrollPosition < 0.32f)
132 {
133     autoScrollRect.verticalScrollbar.value = 1f - normalizedScrollPosition;
134 }
135 }else if(normalizedScrollPosition > 0.32f && normalizedScrollPosition < 0.46f)
136 {
137     autoScrollRect.verticalScrollbar.value = 1f - normalizedScrollPosition - 0.1f;
138 }
139 }else if (normalizedScrollPosition > 0.46f && normalizedScrollPosition < 0.53f)
140 {
141     autoScrollRect.verticalScrollbar.value = 1f - normalizedScrollPosition - 0.18f;
142 }
143 {
144     autoScrollRect.verticalScrollbar.value = 0f;
145 }
146
147 }
148
149 }
```

Listato 5.3: Codice sorgente dello script *ScriptManager*

Di seguito verrà descritto il codice dettagliatamente:

- La prima parte è dedicata all'importazione delle librerie necessarie per il progetto, come quelle dedicate alla gestione delle interfacce utente, del tempo e della connessione a un database MongoDB, e altro.
- La seconda parte si occupa di dichiarare le variabili che memorizzano i dati che verranno utilizzati nel codice, tra cui oggetti *GameObject*, stringhe, liste e dizionari. Ad esempio, *scriptRedListed* e *scriptBlackListed* sono liste di stringhe, *scriptDict* è un dizionario di interi e stringhe, *autoScrollRect* è un riferimento a un oggetto *RectTransform* nell'interfaccia utente, e *txtReview* è un riferimento a un oggetto di testo.
- La funzione *Start()* viene chiamata quando il gioco inizia. Al suo interno, vengono eseguite le seguenti operazioni:
 - Viene richiamata la funzione *GetDataAsync* del componente *mongoDBConnector* per ottenere dati da un database MongoDB.
 - I dati ottenuti dal database vengono memorizzati in un dizionario chiamato *scriptDict*.

- Viene creato un elenco *scriptList* che contiene i valori del dizionario *scriptDict*.
 - Il testo visualizzato nell’oggetto *txtReview* viene impostato utilizzando *string.Join* per concatenare i valori dell’elenco *scriptList*.
 - Viene controllato se *autoScrollRect* è nullo e, in caso affermativo, viene assegnato il componente *ScrollRect* dell’oggetto corrente.
- Il metodo *Update()* viene chiamato ad ogni frame e permette l’aggiornamento continuo della porzione di testo effettivamente letta dal sommelier nell’audio in esecuzione; esso si occupa anche di modificare lo scorrimento del componente *ScrollRect* in base alla posizione in tempo reale dello script in riproduzione. In altre parole, l’obiettivo di questo metodo è garantire che il testo evidenziato sia sempre visibile all’utente durante la riproduzione dell’audio, garantendo un’esperienza simile al karaoke, dove il testo segue la traccia audio in tempo reale.

In particolare:

- La variabile *currentTime* viene aggiornata con il tempo corrente dell’audio ottenuto da un oggetto chiamato *SoundManager*. Il metodo *getCurrentTime()* è utilizzato per recuperare il tempo corrente dell’audio in formato "minuti:secondi".
- Il tempo ottenuto viene suddiviso in minuti e secondi, e questi valori vengono memorizzati nelle variabili *script_minutes* e *script_seconds*.
- Il tempo totale dell’audio viene calcolato convertendo i minuti in secondi e sommandoli tra di loro. Questo valore viene memorizzato in *totalSeconds*.
- L’elenco *scriptRedListed* contiene il testo che deve essere visualizzato fino a quel momento nell’audio. Viene filtrato il dizionario *scriptDict* in modo da includere solo le voci con un tempo chiave minore o uguale a *totalSeconds*. Questo elenco conterrà il testo che deve essere visualizzato in evidenza.
- L’elenco *scriptBlackListed* contiene il testo che non è ancora stato sincronizzato, ovvero il testo successivo a *totalSeconds*. Viene filtrato il dizionario *scriptDict* in modo da includere solo le voci con un tempo chiave maggiore di *totalSeconds*.
- Il testo visualizzato nell’oggetto *txtReview* viene aggiornato. Il testo evidenziato in *scriptRedListed* viene reso rosso (<color=#8F1338>) mentre il testo non ancora sincronizzato in *scriptBlackListed* rimane nel suo colore predefinito.
- Viene calcolata la posizione in pixel del testo evidenziato all’interno del *txtReview*, tenendo conto dell’altezza del testo e del numero di righe evidenziate.
- Viene calcolata la posizione ideale per lo scorrimento verticale in base alla posizione del testo evidenziato rispetto alla vista dell’*autoScrollRect*. Questo consente di posizionare la vista in modo che il testo evidenziato sia visibile all’utente.
- Sulla base della posizione calcolata, la barra di scorrimento verticale dell’*autoScrollRect* viene regolata in modo che l’utente possa vedere il testo evidenziato senza doverlo scorrere manualmente.

5.2.4 Implementazione della classe MongoDBConnector

Questa classe consente di effettuare richieste al servizio MongoDB e di ricevere dati in base a diversi tipi di query consentendo, così, di accedere a informazioni specifiche in base alle esigenze del gioco o dell’applicazione Unity. Nel Listato 5.4 è mostrato il codice della classe.

```

1  using UnityEngine;
2  using UnityEngine.Networking;
3  using System;
4  using System.Collections.Generic;
5  using Newtonsoft.Json;
6  using UnityEngine.UI;
7
8  [Serializable]
9  public class Document
10 {
11     public string Titolo1;
12     public string Titolo2;
13     public float Latitudine;
14     public float Longitudine;
15     public Dictionary<string, string> Descrizione;
16     public string LuogoProd;
17     public Dictionary<string, Dictionary<string, string>> ScriptSommelier;
18 }
19
20
21 [Serializable]
22 public class JsonResponse
23 {
24     public Document document;
25 }
26
27 public class MongoDBConnector : MonoBehaviour
28 {
29     private const string apiKey = "lCcATJK57juG3SHbs16wDjaeniiK41kqdcQt51evKg4yGbGi0sVbegi47Lcw4MK1";
30     private const string url = "https://us-east-1.aws.data.mongodb-api.com/app/data-wftpe/endpoint/data/v1/
31     action/findOne";
32     private string jsonData = "";
33
34     private Action<JsonResponse> onDataReceived;
35
36     public void GetDataAsync(Action<JsonResponse> callback, string queryType)
37     {
38         onDataReceived = callback;
39
40         if (string.Compare(queryType, "all") == 0)
41         {
42             // Create a JSON object with your request data
43             jsonData = "{\"collection\":\"Cantina\", \"database\":\"WineTech\", \"dataSource\":\"ClusterWineTech\", \"_
44             projection\":{\"filter\":{\"Nome\":\"" + ObjectTrackingHandler.GetTrackedObjectName() + "\"}}";
45
46         } else if (string.Compare(queryType, "lat_long") == 0)
47         {
48             jsonData = "{\"collection\":\"Cantina\", \"database\":\"WineTech\", \"dataSource\":\"ClusterWineTech\", \"_
49             projection\":{\"Latitudine\": 1, \"Longitudine\": 1}, \"filter\":{\"Nome\":\"" + ObjectTrackingHandler.
50             GetTrackedObjectName() + "\"}}";
51
52         } else if (string.Compare(queryType, "tit1_tit2_luogo_descr") == 0)
53         {
54             jsonData = "{\"collection\":\"Cantina\", \"database\":\"WineTech\", \"dataSource\":\"ClusterWineTech\", \"_
55             projection\":{\"Titolo1\": 1, \"Titolo2\": 1, \"LuogoProd\": 1, \"Descrizione\": 1}, \"filter\":{\"Nome
56             \"\" + ObjectTrackingHandler.GetTrackedObjectName() + "\"}}";
57
58         } else if (string.Compare(queryType, "tit1_tit2_luogo") == 0)
59         {
60             jsonData = "{\"collection\":\"Cantina\", \"database\":\"WineTech\", \"dataSource\":\"ClusterWineTech\", \"_
61             projection\":{\"Titolo1\": 1, \"Titolo2\": 1, \"LuogoProd\": 1}, \"filter\":{\"Nome\":\"" + ObjectTrackingHandler.
62             GetTrackedObjectName() + "\"}}";
63
64         } else if (string.Compare(queryType, "script") == 0)
65         {
66             jsonData = "{\"collection\":\"Cantina\", \"database\":\"WineTech\", \"dataSource\":\"ClusterWineTech\", \"_
67             projection\":{\"ScriptSommelier\": 1}, \"filter\":{\"Nome\":\"" + ObjectTrackingHandler.
68             GetTrackedObjectName() + "\"}}";
69
70         } else if (string.Compare(queryType, "tit1_tit2_descr") == 0)
71         {
72             jsonData = "{\"collection\":\"Cantina\", \"database\":\"WineTech\", \"dataSource\":\"ClusterWineTech\", \"_
73             projection\":{\"Titolo1\": 1, \"Titolo2\": 1}, \"filter\":{\"Nome\":\"" + ObjectTrackingHandler.
74             GetTrackedObjectName() + "\"}}";
75
76         }
77
78     }
79
80     public void OnDataReceived(JsonResponse response)
81     {
82         onDataReceived(response);
83     }
84
85     public void OnError(string error)
86     {
87         Debug.LogError(error);
88     }
89
90     public void OnProgress(int progress)
91     {
92         Debug.Log("Progress: " + progress);
93     }
94
95     public void OnComplete()
96     {
97         Debug.Log("Operation completed");
98     }
99
100 }
```

```

62     {
63         jsonData = "{\"collection\":\"Cantina\",\"database\":\"WineTech\",\"dataSource\":\"ClusterWineTech\",\""
64             + "projection\":{\"Titolo1\": 1, \"Titolo2\": 1, \"Descrizione\": 1},\"filter\":{\"Nome\":\"" + 
65             ObjectTrackingHandler.GetTrackedObjectName() + "\"}}";
66     }
67     else
68     {
69         // Create a JSON object with your request data
70         jsonData = "{\"collection\":\"Cantina\",\"database\":\"WineTech\",\"dataSource\":\"ClusterWineTech\",\""
71             + "projection\":{},\"filter\":{\"Nome\":\"" + ObjectTrackingHandler.GetTrackedObjectName() + "\"}}";
72     }
73
74     // Create a UnityWebRequest
75     UnityWebRequest request = new UnityWebRequest(url, "POST");
76     request.SetRequestHeader("Content-Type", "application/json");
77     request.SetRequestHeader("Access-Control-Request-Headers", "*");
78     request.SetRequestHeader("api-key", apiKey);
79
80     // Attach the request data
81     byte[] bodyRaw = System.Text.Encoding.UTF8.GetBytes(jsonData);
82     request.uploadHandler = (UploadHandler)new UploadHandlerRaw(bodyRaw);
83     request.downloadHandler = (DownloadHandler)new DownloadHandlerBuffer();
84
85     // Send the request asynchronously
86     request.SendWebRequest().completed += (op) =>
87     {
88         // Handle the response when the request is complete
89         if (request.result == UnityWebRequest.Result.Success)
90         {
91             // Parse the JSON response into a DocumentData object
92
93             JsonResponse data = JsonConvert.DeserializeObject<JsonResponse>(request.downloadHandler.text);
94
95             onDataReceived?.Invoke(data); // Invoke the callback with the data
96         }
97         else
98         {
99             Debug.LogError("Request failed: " + request.error);
100            onDataReceived?.Invoke(null); // Invoke the callback with null to indicate an error
101        }
102    };
103 }

```

Listato 5.4: Codice sorgente dello script *MongoDBConnector*

Di seguito verrà descritto il codice dettagliatamente:

- La prima parte è dedicata all'importazione delle librerie fondamentali per la classe come:
 - *UnityEngine* e *UnityEngine.Networking*, che consentono di utilizzare le funzionalità di Unity per la gestione dell'ambiente di gioco e le richieste di rete.
 - *System.Collections.Generic*, che consente di utilizzare le collezioni di dati, come il dizionario.
 - *Newtonsoft.Json*, che consente la serializzazione e deserializzazione dei dati JSON.
 - *UnityEngine.UI*, che permette la gestione degli elementi dell'interfaccia grafica all'interno di Unity.
- La seconda parte è dedicata alla definizione della classe utili per la deserializzazione dei dati provenienti dal documento MongoDB, ovvero:
 - La classe *Document* contiene vari campi, come *Titolo1*, *Titolo2*, *Latitudine*, *Longitudine*, *Descrizione*, *LuogoProd* e *ScriptSommelier*, ognuno dei quali è associato a un tipo di dato specifico appartenente al documento MongoDB.

- La classe *JsonResponse* rappresenta la struttura dei dati JSON restituiti dalla richiesta al servizio MongoDB. Essa contiene un campo *document* che è un oggetto di tipo *Document*.
- La terza parte del codice è dedicata alla dichiarazione degli attributi tra cui:
 - *apiKey*: contiene l'API key del servizio MongoDB a cui si sta cercando di accedere.
 - *url*: è una stringa che contiene l'URL del servizio MongoDB a cui si sta cercando di accedere.
 - *jsonData*: è una variabile stringa che conterrà i dati JSON della richiesta.
 - *onDataReceived* è una variabile di tipo *Action<JsonResponse>* che verrà utilizzata per passare una funzione di callback per elaborare i dati ricevuti.
- L'ultima parte del codice definisce il metodo *GetDataAsync* che viene utilizzato per effettuare una richiesta asincrona al servizio MongoDB. Esso riceve due parametri ovvero *callback* (la funzione di callback) e *queryType* (il tipo di query da eseguire). In base al valore di *queryType*, viene costruita una stringa JSON (*jsonData*) che rappresenta la richiesta da inviare al servizio MongoDB. Viene creato un oggetto *UnityWebRequest* per gestire la richiesta HTTP POST. Vengono impostati gli header e il corpo della richiesta. La richiesta viene inviata in modo asincrono e un callback gestisce la risposta. Se la richiesta ha successo, i dati JSON ricevuti vengono deserializzati in un oggetto *JsonResponse* e passati alla funzione di callback *onDataReceived*. In caso di errore, viene generato un messaggio e la funzione di callback viene chiamata con *null* per indicare un errore.

5.3 Implementazione delle animazioni in Unity

In questa sezione verranno riportate le implementazioni delle due animazioni principali presenti nell'applicazione Android. In particolare verranno descritte quelle presenti nel menù principale e nella pagina dedicata alla selezione dell'annata.

5.3.1 Menù principale

Il menù principale, come ampiamente descritto nel capitolo precedente, presenta un'animazione che fornisce i primi pulsanti dedicati alla selezione, rispettivamente, delle tre sezioni "Osserva", "Ascolta" e "Racconta". In Figura 5.5 è mostrata la sequenza di operazioni compiute dall'animazione *MenuFadeIn* all'avvio dell'applicazione.

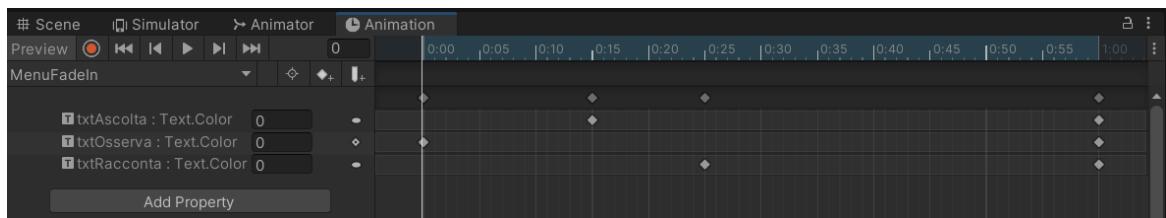


Figura 5.5: Sequenza di operazioni svolte all'avvio dell'applicazione dall'animazione *MenuFadeIn*

L'animazione è molto semplice: nel momento in cui l'applicazione è avviata, attiva un *fadeIn* dei tre pulsanti "Osserva", "Ascolta" e "Racconta". Per creare l'effetto di dissolvenza in ingresso, viene modificata la componente *alpha* del colore del testo dei pulsanti *txtOsserva*,

txtAscolta e *txtRacconta* portandolo da un valore iniziale di 0 fino ad 1. Ogni transizione ha un differente punto di partenza per creare un effetto gradevole all’utente.

In Figura 5.6 è fornita un’immagine della sequenza di operazioni che viene svolta nel momento in cui viene premuto il pulsante "Osserva". Per gli altri due pulsanti (Racconta e Ascolta), la sequenza di operazioni svolte dall’animazione è molto simile a quella descritta in seguito e quindi non verranno descritte.

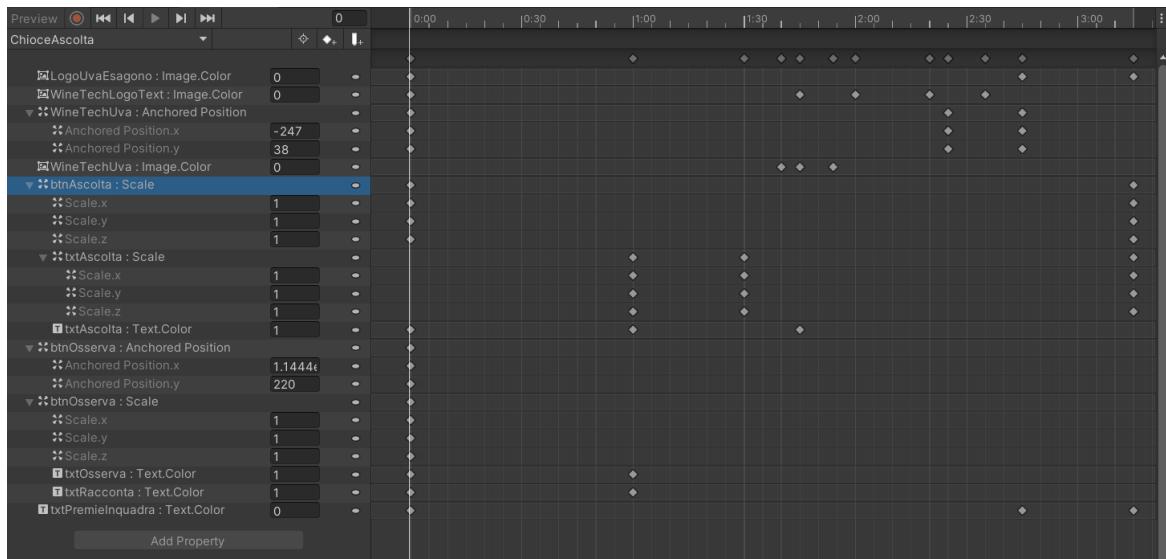


Figura 5.6: Sequenza di operazioni svolte dall’animazione *ChoiceOsserva*

Per prima cosa, nel momento in cui l’utente preme il pulsante "Osserva", viene effettuato un effetto di *fadeOut* dei due pulsanti "Ascolta" e "Racconta", portando il valore di *alpha* da 1 a 0 in maniera graduale.

Subito dopo, viene effettuata una traslazione del pulsante "Osserva" dall’alto verso il centro modificando il parametro *Anchored Position.y* dal valore 220 a 68 in maniera progressiva.

Successivamente viene aumentata gradualmente la grandezza del pulsante portando i valori di *Scale.x*, *Scale.y* e *Scale.z* da 1 ad 1.2, sempre in maniera graduale. Nel frattempo, viene inizializzata anche la transizione di *fadeOut* del pulsante "Osserva" che porta il parametro *alpha* del pulsante da 1 a 0.

a questo punto vengono mostrate in modo temporalmente sfalsato, con un effetto *fadeIn*, il logo dell’applicazione e la scritta "WineTech" (rispettivamente chiamati *WineTechUva* e *WineTechLogoText*), modificandone il parametro *alpha* da 0 a 1 .

Dopo appena un secondo, viene attivato il *fadeOut* del pulsante *WineTechLogoText* modificando il parametro *alpha* da 1 a 0. Contemporaneamente viene effettuata una traslazione verso destra del logo dell’applicazione modificando il valore *Anchored Position.x* da -245.19 a 2. Quasi in contemporanea, viene effettuato un *fadeIn* del pulsante a forma di esagono che permette la transizione alla scena successiva, sempre modificando il parametro *alpha* da 0 a 1 in maniera graduale.

5.3.2 Selezione dell’annata

Per quanto riguarda la schermata che permette la selezione dell’annata, l’animazione è più semplice rispetto a quella del menù principale. Infatti l’animazione coinvolta, denominata *Scene2Intro*, effettua una modifica del parametro *Anchored Position.y* dal valore -1080 a 0 per permettere lo scorrimento dal basso verso l’alto del pannello di selezione dell’annata. Successivamente, nel momento in cui la selezione è stata effettuata, viene ulteriormente

incrementato il valore di *Anchored Position.y* portandolo al valore di 0, ma stavolta del componente *Background* dell’interfaccia grafica della scena.

5.4 Manuale utente

5.4.1 Premessa

Normalmente, un’applicazione Android è presente nel Google Play Store da cui è possibile scaricarla ed installarla in pochi minuti. Purtroppo quest’applicazione essendo un prototipo, non è ancora presente in questa piattaforma. Questa è la ragione per cui necessita di alcuni passaggi aggiuntivi per essere installata correttamente. In futuro, se il progetto verrà portato a compimento, l’applicazione sarà caricata all’interno del Google Play Store per essere aumentare l’accessibilità del prodotto. Tramite Unity, è possibile ottenere il file APK che consente l’installazione dell’applicazione una volta copiato il file all’interno dello smartphone. Bisogna, pertanto, avere l’autorizzazione da parte dell’utente per installare nuove applicazioni non certificate e quindi considerate pericolose.

5.4.2 Effettuare il riconoscimento di un prodotto

Una volta avviata l’applicazione, al primo utilizzo potrebbe essere richiesta l’autorizzazione per accedere alla fotocamera, che è essenziale per il riconoscimento dei prodotti vitivinicoli.

Dopo l’apertura dell’applicazione e l’apparizione dei pulsanti “Osserva,” “Ascolta” e “Racconta” sullo schermo, è sufficiente premere il pulsante corrispondente alla sezione di interesse. Questa azione attiverà le animazioni e porterà alla visualizzazione del pulsante a forma di esagono. Quando si è in grado di avviare il riconoscimento del prodotto vitivincolo, è possibile premere il pulsante per accedere alla schermata dedicata al riconoscimento.

L’operazione di riconoscimento avviene in modo automatico e se il prodotto viene identificato con successo, l’applicazione passerà alla schermata successiva che consente la selezione dell’annata del prodotto vitivincolo. È sufficiente scegliere quest’ultima nel campo apposito e premere poi il pulsante “Avanti.” A questo punto, l’utente sarà riportato alla schermata relativa alla scelta effettuata nelle prime fasi dell’applicazione.

5.4.3 Utilizzo delle sezioni principali

Sezione Osserva

La sezione *Osserva* presenta una pagina singola con alcune informazioni sul prodotto vitivincolo riconosciuto. L’utente può scorrere verso il basso per visualizzare il video dedicato all’azienda vitivinicola che viene riprodotto automaticamente.

Sezione Ascolta

Per quanto riguarda la sezione *Ascolta*, basta attendere che il caricamento dello script sia effettuato da parte dell’applicazione. Una volta ottenuto il testo della recensione del sommelier, è sufficiente premere il pulsante *Play* e, quindi, riprodurre l’audio relativo al prodotto vitivincolo. Per tornare indietro o per andare avanti nella riproduzione, è sufficiente trascinare la barra di riproduzione verso destra o sinistra.

Sezione Racconta

La sezione *Racconta*, invece, visualizza le informazioni in tempo reale associate al vigneto di provenienza del prodotto vitivinicolo. I dati vengono richiesti nel momento in cui viene attivata la sezione. Se si desidera aggiornare i dati relativi alla stazione IoT o quelli relativi a *OpenWeather*, è possibile premere i relativi pulsanti presenti nell'interfaccia grafica.

CAPITOLO 6

Confronto con sistemi di tracciamento

6.1 Aspetti in comune del progetto con i sistemi di tracciamento

6.2 Aspetti differenti del progetto con i sistemi di tracciamento

CAPITOLO 7

Discussione in merito al lavoro svolto

7.1 Confronto con l'app Vivino

7.2 SWOT Analysis

7.3 Sviluppi futuri

CAPITOLO 8

Conclusioni

Bibliografia

PATEL, G. S. (2021), *Smart agriculture emerging pedagogies of deep learning, Machine Learning and internet of things*, CRC Press.

ZHANG, Q. (2016), *A History of Precision Agriculture*, p. 1–13, Taylor & Francis.

Siti web consultati

- Unity Real-Time Development Platform | 3D, 2D, VR & AR Engine – www.unity.com;
- Vuforia Engine Developer Portal <https://developer.vuforia.com>;
- Polycam - LiDAR & 3D Scanner for iPhone & Android <https://poly.cam>;
- blender.org - Home of the Blender project - Free and Open 3D Creation Software <https://www.blender.org>;
- Google Earth Studio <https://www.google.com/intl/it/earth/studio/>;
- MongoDB: La Piattaforma Di Dati Applicativi | MongoDB <https://www.mongodb.com/it-it>;

Ringraziamenti
