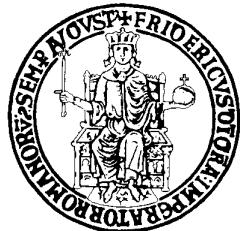


**UNIVERSITÀ DEGLI STUDI DI NAPOLI
“FEDERICO II”**



**DIPARTIMENTO DI INGEGNERIA ELETTRICA E
TECNOLOGIE DELL'INFORMAZIONE**

Corso di Laurea in Informatica

**SPIEGARE LE RISPOSTE DEI SISTEMI DI MACHINE
LEARNING: UN APPROCCIO A SCATOLA NERA PER LE
CLASSIFICAZIONI**

Relatore

Ch.mo Prof. Roberto Prevete

Correlatore

Ch.ma Prof.ssa Anna Corazza

Candidato

Andrea Sorrentino

Matr. N97/221

Anno Accademico 2017 – 2018

*Alla mia famiglia.
Ai miei amici più cari.*

Indice

Introduzione	1
1 Interpretabilità dei sistemi di Machine Learning	5
1.1 Importanza dei sistemi autonomi basati su tecniche di Machine Learning	6
1.2 Importanza dell'interpretabilità nei sistemi autonomi	8
1.3 Il ruolo della legge	14
1.3.1 GDPR - General Data Protection Regulation	14
1.4 Interpretabilità	16
1.4.1 Trasparenza	18
1.4.2 Interpretabilità post-hoc	19
1.4.3 Valutare i metodi per l'interpretabilità	20
1.5 Spiegazione	21
1.5.1 Quando generare una spiegazione	23
1.6 Direzione in cui si sviluppa l'elaborato	24
2 Lo stato dell'arte	28
2.1 Una tassonomia condivisa	29
2.2 Interpretabilità globale e locale	32
2.2.1 Activation Maximization (AM)	34
2.2.2 Sensitivity Analysis	41
2.2.3 Local Interpretable Model-agnostic Explanations - LIME .	43
3 Un nuovo approccio al problema dell'interpretabilità	48
3.1 L'idea di base e l'architettura del sistema	48

3.1.1	Fase 1 - Apprendimento del dizionario mediante Dictionary Learning	53
3.1.2	Fase 2 - Selezione degli elementi strutturali necessari alla spiegazione dell’istanza	55
3.2	Scelta della tecnica di Dictionary Learning	58
3.2.1	NMF (Non-negative Matrix Factorization)	59
3.2.2	NMF con vincoli di sparsità	61
3.2.3	Implementazione della nuova tecnica	67
4	Test e Risultati	69
4.1	Descrizione dei dataset	69
4.1.1	MNIST	70
4.1.2	Fashion-MNIST	71
4.2	Descrizione del sistema di classificazione utilizzato: LeNet-5	72
4.3	Test della fase 1: generazione dei dizionari	75
4.3.1	Test dei dizionari su MNIST	75
4.3.2	Test dei dizionari su Fashion-MNIST	79
4.4	Test della fase 2: generazione della spiegazione	86
4.4.1	Test sull’interpretabilità locale	86
4.4.2	Test sull’interpretabilità globale	95
Conclusioni e sviluppi futuri	101	
Bibliografia	109	

Elenco delle figure

1.1	Esempio di percezione dell'environment in un sistema a guida autonoma. Immagine ottenuta da https://www.nvidia.com/en-us/gtc/topics/autonomous-driving/	8
1.2	Immagine ottenuta da <i>Explainable Artificial Intelligence (XAI)</i> - <i>David Gunning - DARPA/I20</i>	9
1.3	Immagine ottenuta da [Montavon et al., 2017]. Mostra l'uso di alcune tecniche di spiegazione di modelli complessi per l'analisi di dati scientifici	12
1.4	Modello di una rete neurale a strati full connected	13
1.5	Architettura del modello GoogLeNet	13
1.6	GDPR - Articolo 22	15
1.7	Immagine tratta da [Wasserman, 2010]. Mostra un classificatore lineare su dati rappresentati dalle due variabili x_1 e x_2	26
2.1	Numero di paper pubblicati per anno riguardanti la MLI ¹	28
2.2	Esempi di attribution methods, ottenuti da [Montavon et al., 2017]	31
2.3	Immagine tratta da [Erhan et al., 2009] AM applicato a MNIST. Visualizzazione di 36 unità dal primo, secondo e terzo strato interno di una Deep Belief Network.	34
2.4	Prototipi per MNIST da 0 (a sinistra) a 9 (a destra), generati mediante AM, su Lenet-5 [Lecun et al., 1998], con la quantità di interesse proposta in 3.1	35
2.5	Iterazioni di AM nel corso degli anni, immagine ottenuta da [Olah et al., 2017]	36

2.6	Esempio di AM con regolarizzazione $\lambda x ^2$, su ConvNet, addestrato su ILSVRC-2013. Immagine tratta da [Simonyan et al., 2013]	37
2.7	Esempio di AM con l'utilizzo delle GAN, immagine tratta da [Nguyen et al., 2016a]	38
2.8	Prototipi per MNIST, generati mediante AM, su Lenet-5 [Lecun et al., 1998], con inserimento della media nella regolarizzazione	39
2.9	Tecnica e risultato della stessa come mostrati in [Nguyen et al., 2016b]	40
2.10	Immagine tratta da [Montavon et al., 2017]. A sinistra l'applicazione della sensitivity analisys ad una DNN per la spiegazione di un'immagine rappresentante una barca. A destra la stessa tecnica applicata ad una CNN sui digit di MNIST.	41
2.11	Immagine tratta da [Kindermans et al., 2017a]. Valutazione di vari attribution method per la sensitivity analysys su MNIST, su due modelli con codifiche diverse. Il primo (f1) nell'intervallo [0,1], il secondo (f2) nell'intervallo [-1,0]. I metodi Gradient x Input, IG e LRP non sono affidabili e producono risultati diversi a seconda della rete. IG con un black reference point e PatternAttribution sono invarianti rispetto alla trasformazione dell'input.	43
2.12	Immagine tratta da [Ribeiro et al., 2016]. Esempio dummy di un modello lineare appreso su un dataset di sample perturbati attorno all'istanza da spiegare. Localmente un modello lineare riesce ad approssimarne uno complesso.	44
2.13	Immagine tratta da [Ribeiro et al., 2016]. Esempio di spiegazione per le Classi: Labrador, chitarra classica e chitarra elettrica	44
2.14	Immagine tratta da [Ribeiro et al., 2016]. Algoritmo principale per la generazione della spiegazione mediante LIME	46
2.15	Immagine tratta da [Ribeiro et al., 2016]. SP-LIME: algoritmo per migliorare l'interpretabilità globale di un modello.	47
3.1	Esempio di feature in un documento, a sinistra. A destra esempio di feature in un'immagine.	50

3.2	Idea di base del sistema. Spiegazione del perché il modello in questione ha classificato l'immagine come un 3.	51
3.3	Architettura del sistema. A sinistra un framework per l'interpretabilità locale. A destra lo stesso framework per l'interpretabilità globale.	52
3.4	Fase 1 della tecnica sviluppata, generazione di un dizionario di strutture di base, dette atomi.	53
3.5	Rappresentazione grafica di una tecnica di dictionary learning. V rappresenta il dataset di partenza, costituito da T elementi di dimensionalità N . W è il dizionario che si va ad apprendere, formato da K atomi di dimensionalità N . H è l'encoding, ossia la matrice dei pesi necessaria a ricostruire il dataset di partenza, mediate un prodotto vettoriale con il dizionario. H è una matrice di T elementi di dimensionalità K	54
3.6	Fase 2 della tecnica sviluppata. Immagine riassuntiva. Interpretabilità locale.	57
3.7	Esempio di struttura sparsa che intendiamo generare a sinistra. A destra un esempio di sparsità non corretta.	58
3.8	NMF - immagine tratta da [Lee and Sebastian Seung, 1999]. W rappresenta il dizionario, in questo caso gli atomi sono rappresentati in modo visuale. Per questo motivo anziché essere vettori sono immagini. Allo stesso modo h , ossia l'encoding necessario alla ricostruzione dell'immagine originaria è espresso come matrice. . .	60
4.1	Esempi di immagini estratte dal dataset MNIST	70
4.2	Esempi di immagini estratte dal dataset Fashion-MNIST	72
4.3	Immagine tratta da [Lecun et al., 1998]. In figura è mostrato uno schema riassuntivo dell'architettura di LeNet-5.	73

4.4	Dizionari generati mediante NMF con vincoli di sparsità su MNIST. I dizionari sono generati su tutte le classi del dataset. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell'errore di ricostruzione.	76
4.5	Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.	77
4.6	Esempi di atomi estratti in modo random dal dizionario con i valori di lambda ritenuti peggiori.	78
4.7	Dizionari generati mediante NMF con vincoli di sparsità su MNIST. I dizionari sono generati sulla classe 8. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell'errore di ricostruzione.	78
4.8	Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.	79
4.9	Esempi di atomi estratti in modo random dal dizionario con i valori di lambda ritenuti peggiori.	79
4.10	Esempi di atomi estratti da un dizionario generato su Fashion-MNIST in cui è stata applicata la sparsità sia sul dizionario che sull'encoding. In particolare $\lambda_1 = 0.8$ e $\lambda_2 = 0.7$. Una delle migliori combinazioni di parametri per MNIST.	80
4.11	Dizionari generati mediante NMF con vincoli di sparsità su Fashion-MNIST. I dizionari sono generati su tutte le classi del dataset. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell'errore di ricostruzione.	80
4.12	Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.	81
4.13	Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.	82

4.14 Dizionari generati mediante NMF con vincoli di sparsità su Fashion-MNIST. I dizionari sono generati su sulla classe 'sandalo'. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell'errore di ricostruzione.	83
4.15 Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.	84
4.16 Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti peggiori.	84
4.17 Immagine di un 3 da spiegare.	86
4.18 Spiegazioni generate per l'immagine 3.	87
4.19 Immagine di un 7 da spiegare.	88
4.20 Spiegazioni generate per l'immagine 7.	89
4.21 Generazione della spiegazione 4 volte sulla stessa immagine. Gli atomi selezionati sono leggermente diversi tra loro di volta in volta. Nonostante ciò la spiegazione rimane comprensibile e significativa.	90
4.22 Immagine di un 'sandalo' da spiegare.	91
4.23 Spiegazioni generate per l'immagine 'sandalo'.	92
4.24 Immagine di un 'pullover' da spiegare.	93
4.25 Spiegazioni generate per l'immagine 'sandalo'.	93
4.26 Generazione del prototipo per la classe 9. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.	96
4.27 Generazione del prototipo per la classe 2. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.	96
4.28 Generazione di più prototipi per la classe 5. La variabilità dei prototipi non incide in modo significativo sulla comprensione delle caratteristiche importanti di una classe.	97
4.29 Generazione del prototipo per la classe 'pullover'. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.	98
4.30 Generazione del prototipo per la classe 't-shirt'. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.	99

Introduzione

Nell'ultimo decennio, la diffusione di sistemi decisionali autonomi è aumentata in modo esponenziale. Il successo di queste tecniche è dovuto per lo più allo sviluppo di metodologie avanzate nella disciplina del Machine Learning. Comprensione più profonda dei processi di apprendimento automatico, maggiore capacità computazionale e un'enorme mole di dati a disposizione sono solo alcuni dei fattori che hanno scatenato la corsa “all'intelligenza artificiale”.

I sistemi in questione sono estremamente popolari e stanno iniziando a pervadere aree critiche come la sanità, la finanza e i trasporti, raggiungendo, in alcuni task specifici, prestazioni che superano quelle degli esseri umani [Doshi-Velez and Kim, 2017]. Per analizzare la portata di questa rivoluzione, basta prendere in considerazione alcuni eventi chiave tra i tanti avvenimenti che si sono succeduti nel corso degli anni:

- Nel 2014 Facebook ha pubblicato un lavoro di ricerca sul riconoscimento facciale detto DeepFace, il sistema usa un modello in grado di identificare i volti con un'accuratezza del 97.35%, mai raggiunta in precedenza¹.
- Nel 2015 il sistema AlphaGo, messo a punto da Google, è stato in grado di sconfiggere per la prima volta un giocatore professionista di Go. Considerato da molti il miglior giocatore della storia².
- Nel 2016, la Oxford University, in collaborazione con Google, ha messo a punto un sistema in grado di estrarre il testo dal labiale delle perso-

¹<https://research.fb.com/publications/deepface-closing-the-gap-to-human-level-performance-in-face-verification/>

²<https://deepmind.com/research/alphago/>

ne, con una precisione superiore a quella di un professionista del settore [Chung et al., 2016].

- Nel 2016 Google ha utilizzato i risultati del progetto DeepMind per addestrare un modello al fine di ottimizzare gli impianti di raffreddamento dei propri datacenter. In questo modo è riuscita a ridurre le spese per il raffreddamento dei sistemi del 40% ³.
- Nel 2016 Waymo ha suggellato un accordo commerciale con FCA per produrre la prima auto a guida autonoma acquistabile negli Stati Uniti⁴.
- Nel 2017 Google ha messo a punto un sistema con l'obiettivo di aiutare un Patologo ad individuare il cancro nei pazienti, tramite il Deep Learning⁵.

Nonostante i successi ottenuti, questi sistemi non sono privi di errori ed inconsistenze, spesso sottovalutati dalla comunità scientifica. Infatti, la complessità dei modelli di Machine Learning e le loro enormi abilità predittive, li portano ad essere, spesso, sfruttati come black box, al punto che [Pasquale, 2015] introduce il termine “Black Box Society”. Proprio per questo, l’incapacità degli addetti ai lavori di capire totalmente tali modelli diventa una problematica importante [Lipton, 2016], che va affrontata in modo strutturato e sotto vari punti di vista (etico, legislativo, etc.). La difficoltà di comprendere il ragionamento logico alla base della predizione di un modello è una questione di primaria importanza. Questo, soprattutto, perché tali sistemi sono addestrati su enormi dataset, spesso ottenuti a partire da tracce digitali di attività umane, e proprio per questo potrebbero sfruttare dei bias presenti nei dati, ereditando così pregiudizi ed artefatti propri dei comportamenti umani (e.g. razzismo verso le minoranze) [Danks and London, 2017]. In definitiva, facendo uso di modelli di Machine Learning che non riusciamo a comprendere appieno, rischiamo di creare sistemi con un forte impatto dal punto di vista etico, della sicurezza e della responsabilità,

³<https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/>

⁴<https://waymo.com/>

⁵<https://ai.googleblog.com/2017/03/assisting-pathologists-in-detecting.html>

senza avere poi la capacità di controllarli in modo corretto [Ras et al., 2018]. Nell’elaborato saranno quindi analizzate delle tecniche che rispondono ai problemi che abbiamo evidenziato e, sulla base del lavoro già presente in letteratura, sarà proposta una nuova metodologia che affronta tali questioni da un punto di vista innovativo. La dissertazione sarà strutturata nel seguente modo.

Nel *Capitolo 1* esporremo, innanzitutto, dei concetti utili ad impostare una discussione critica sull’importanza dei sistemi autonomi basati su tecniche di Machine Learning. Successivamente, analizzeremo dei casi di fallimento, che ci aiuteranno a comprendere come sia necessario intervenire al più presto per la regolamentazione dell’utilizzo di tali sistemi. Infatti, come affermato in [Doshi-Velez et al., 2017] “Siamo in un momento storico per cui delle scelte giuste, in campo legislativo, sul quando richiedere una spiegazione potrebbero aiutare a prevenire scenari negativi per i sistemi di AI, mentre delle decisioni sbagliate non solo fallirebbero ad attribuire la giusta responsabilità ai sistemi di AI, ma arresterebbero lo sviluppo di tecnologie da cui l’uomo potrebbe trarre grande beneficio”. Pertanto, ci focalizzeremo sullo studio del GDPR (General Data Protection Regulation) [Goodman and Flaxman, 2017], una direttiva europea, emanata nel 2016, con l’obiettivo di regolamentare l’utilizzo di sistemi in grado di prendere decisioni in modo autonomo. In particolare ci soffermeremo sulla norma inerente l’*“Automated individual decision-making, including profiling”* e sul *“diritto alla spiegazione”* che deriva dall’applicazione della stessa. Mostreremo come la spiegazione cui si fa riferimento nel GDPR sia solo uno degli strumenti atti ad aumentare l’interpretabilità del sistema. A tal proposito, introdurremo il concetto di *interpretabilità* e mostreremo come esso sia di fondamentale importanza per giudicare la bontà di un modello, se affiancato alle usuali misure prestazionali (e.g. accuratezza). Porteremo, quindi, delle critiche al concetto di interpretabilità che, sebbene sia spesso usato in senso quasi matematico [Lipton, 2016], risulta per lo più ambiguo e sfuggente. Apriremo quindi un’ampia parentesi sulle spiegazioni, strumento fondamentale al nostro senso di comprensione. Infine, chiariremo la direzione in cui intendiamo operare al fine di sviluppare una nuova metodologia che risolva il

problema dell’interpretabilità dei sistemi autonomi basati su tecniche di Machine Learning.

Nel *Capitolo 2* esamineremo lo stato dell’arte. Nello specifico, faremo chiarezza sull’insieme di tecniche presenti in letteratura e proporremo una tassonomia standard [Ras et al., 2018] per le stesse. Infine, analizzeremo in modo approfondito tre tecniche: Activation Maximization, Sensitivity Analysis e LIME. La comprensione di tali strumenti sarà di grande aiuto nello sviluppo di un approccio alternativo al problema in questione.

Nel *Capitolo 3* esporremo l’idea alla base della tecnica che intendiamo sviluppare. Mostreremo come il nostro approccio estrapoli i punti di forza dei metodi analizzati al fine di creare una tecnica più robusta. Imposteremo l’architettura del sistema e comprenderemo meglio come integrare le varie parti dello stesso. A questo punto, sarà di fondamentale importanza l’analisi di una di queste componenti: il dictionary learning. Esporremo quelli che, a nostro avviso, devono essere i principi sulla cui base scegliere una tecnica di dictionary learning adeguata alla metodologia che proponiamo. Effettueremo una scelta in particolare e la giustificheremo. Passeremo poi all’implementazione effettiva della tecnica e all’analisi delle difficoltà incontrate durante la specifica della stessa.

Nel *Capitolo 4* presenteremo i dataset su cui saranno effettuati i test. Seguendo delle metodologie standard, metteremo alla prova le tecniche per la generazione di dizionari scelti nel *Capitolo 3* e ricercheremo i punti di forza e di debolezza delle stesse. Utilizzeremo quindi l’algoritmo proposto nel terzo capitolo per generare delle spiegazioni ed in modo qualitativo comprenderemo come esse possano aiutarci a risolvere il problema dell’interpretabilità nei sistemi decisionali autonomi basati sul Machine Learning.

Infine, trarremo le conclusioni a margine del lavoro svolto e proporremo degli sviluppi futuri atti a migliorare il sistema.

Capitolo 1

Interpretabilità dei sistemi di Machine Learning

Il contenuto del seguente capitolo pone le basi per una discussione critica sul ruolo dell'interpretabilità nel Machine Learning. Nello specifico sarà analizzata, inizialmente, l'importanza dei sistemi automatici ed il ruolo che ha il Machine Learning nella loro creazione. Ogni tecnologia nuova porta con sé una serie di interrogativi e di problemi che vanno risolti, pertanto esamineremo tali quesiti ed il loro impatto sulla produzione di sistemi affidabili. A tal proposito prenderemo in considerazione il ruolo che la legge dovrebbe assumere nei suddetti ambiti, soprattutto in relazione ad un regolamento europeo emanato nel 2016 ed entrato in vigore il 25 maggio 2018, il GDPR, o General Data Protection Regulation. Si passerà poi alla definizione ed alla comprensione approfondita dei concetti di interpretabilità e spiegazione. Infine, chiariremo la nostra posizione sull'argomento, i punti su cui andremo ad incentrare la presente dissertazione e i limiti degli approcci analizzati.

1.1 Importanza dei sistemi autonomi basati su tecniche di Machine Learning

Come accennato nel paragrafo introduttivo, negli ultimi decenni la diffusione di sistemi informatici in grado di prendere decisioni in modo “autonomo” è stata rapida e capillare. A livello accademico, sono stati raggiunti livelli prestazionali senza precedenti in svariati campi, tra cui, la Visione Computazionale ed il Natural Language Processing [Ras et al., 2018]. A livello industriale, tali sistemi stanno influenzando molteplici settori, tra cui: automobilistico [Bojarski et al., 2017], legislativo¹ e medico [Lee et al., 2017]. Il successo di questi software è dovuto per lo più all’utilizzo di tecniche di Machine Learning che hanno determinato un notevole incremento prestazionale dei sistemi a disposizione dei professionisti dei vari campi. Tale cambiamento, radicale nel mondo dell’apprendimento automatico, è attribuibile in buona parte a:

- ***Un rapido sviluppo di modelli di Machine Learning complessi.*** Nel corso del tempo sono stati creati modelli sempre più complessi con milioni di parametri, come le Deep Neural Networks (DNN), in grado di raggiungere prestazioni al pari o addirittura superiori agli umani in alcuni settori specifici.
- ***Un netto miglioramento delle tecniche di training.*** La creazione di sistemi complessi è stata resa possibile grazie all’utilizzo di nuove tecniche, ad esempio nell’ambito delle reti neurali: funzioni di attivazione differenti come le ReLU (Rectified Linear Unit), tecniche di regolarizzazione come dropout e batch regularization e migliore comprensione della corretta strutturazione di un modello.
- ***Un incremento delle capacità computazionali.*** L’implementazione degli algoritmi su GPU ha consentito di sfruttare le capacità di parallelismo

¹Alan Lockett, Trevor Jefferies, Neil Etheridge, and Alicia Brewer. White paper tag predictions: How DISCO AI is bringing deep learning to legal technology. <https://www.csdisco.com/disco-ai>

di questi dispositivi ai massimi livelli. Ciò ha portato ad una notevole riduzione dei tempi di apprendimento dei modelli.

- ***Disponibilità di una enorme mole di dati.*** La creazione di dataset enormi, contenenti milioni di immagini etichettate ha permesso di sfruttare al massimo le capacità di apprendimento di questi algoritmi.

- ***Una diffusione di framework software come Tensorflow e Caffe.***

Tali strumenti sono stati in grado di offrire API di alto livello per operazioni complesse e ciò ha fortemente contribuito alla diffusione di questi argomenti presso sviluppatori senza un background di conoscenze matematiche formali.

Come affermato, nel mondo accademico, questo progresso ha significato un balzo in avanti eccezionale in campi di ricerca come la classificazione di immagini e l'object detection, la speech recognition ed il natural language processing. Ciò si è, ovviamente, riflettuto al di là del mondo accademico con un forte impatto sulla società e la produzione industriale [Pedreschi et al., 2018]. Al giorno d'oggi i sistemi di Machine Learning più avanzati hanno raggiunto prestazioni al pari, o superiori, agli esseri umani su task specifici [Doshi-Velez and Kim, 2017]. Si pensi al celebre caso AlphaGo, un sistema di intelligenza artificiale, messo a punto da Google, che ha battuto per la prima volta un giocatore professionista (Lee Sedol) di Go [Silver et al., 2016]. La capacità di mettere in atto dei sistemi del genere porta introiti per milioni di dollari alle aziende e spesso questi soldi sono reinvestiti nello sviluppo di sistemi ancora più avanzati, creando un circolo virtuoso per la ricerca a livello industriale. Basti pensare allo sviluppo di assistenti vocali come Alexa, Siri e Google Duplex. Grazie a questi algoritmi le aziende sono in grado di fornire servizi con un grado di personalizzazione sempre maggiore e riescono a creare dei recommendation systems che si adattano sempre più alle nostre esigenze. In molti casi i software sviluppati hanno come obiettivo quello di migliorare le prestazioni di un essere umano andandole ad integrare con quelle di un sistema di intelligenza artificiale debole, tipici in questo senso sono gli strumenti a disposizione degli automobilisti, messi in campo da aziende come la

Tesla. Tuttavia, così come cresce la presenza di modelli di Machine Learning in campi critici come la medicina e la finanza, allo stesso modo l’incapacità degli addetti ai lavori di comprendere appieno questi sistemi e quindi di controllarli correttamente diventa una questione di massima rilevanza [Lipton, 2016]. Nella prossima sezione affronteremo i problemi derivanti dall’utilizzo di tali sistemi, mettendo in luce come un loro impiego come black-box possa risultare pericoloso.



Figura 1.1: Esempio di percezione dell’environment in un sistema a guida autonoma. Immagine ottenuta da <https://www.nvidia.com/en-us/gtc/topics/autonomous-driving/>

1.2 Importanza dell’interpretabilità nei sistemi autonomi

Nonostante le notevoli capacità prestazionali e la loro diffusione capillare, i modelli di Machine Learning più complessi rimangono per lo più delle black-box, ossia delle scatole nere, impossibili da scrutare. Come affermato chiaramente in [Pedreschi et al., 2018], essi “mappano degli utenti su delle classi, basandosi su delle feature attribuibili agli stessi, senza spiegare il perché della scelta effettuata”². Questo aspetto è decisamente preoccupante e va studiato correttamente. Tali sistemi, infatti, sono addestrati a partire da enormi dataset, spesso ottenuti a partire da tracce digitali di attività umane, e proprio per que-

²Traduzione a cura degli autori del presente elaborato

sto potrebbero sfruttare dei bias presenti nei dati, ereditando così pregiudizi ed artefatti propri dei comportamenti umani (e.g. razzismo verso le minoranze) [Danks and London, 2017]. In definitiva, sfruttando modelli di Machine Learning che non riusciamo a comprendere appieno, rischiamo di creare sistemi con un forte impatto dal punto di vista etico, della sicurezza e della responsabilità, senza avere poi la capacità di controllarli in modo corretto [Pedreschi et al., 2018].

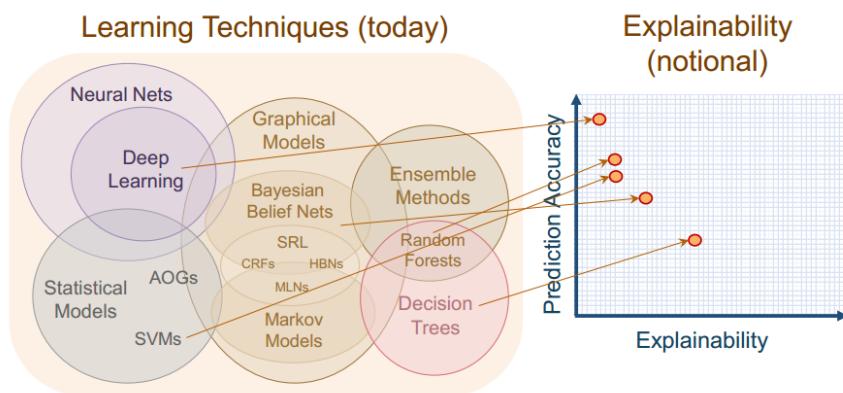


Figura 1.2: Immagine ottenuta da *Explainable Artificial Intelligence (XAI)* - David Gunning - DARPA/I20

Proprio per questo, per gli analisti ed i data scientist, l'enorme capacità predittiva dei sistemi in questione, potrebbe non riuscire a controbilanciare le implicazioni dal punto di vista etico e legislativo (aspetto che sarà affrontato nella *Sezione 1.3*). Una situazione del genere potrebbe costringerli ad utilizzare ancora sistemi il cui grado di interpretabilità è alto, come modelli di regressione lineare/logistica, in cui basta osservare il vettore di pesi per comprendere il motivo della predizione. Come già accennato, questa difficoltà degli addetti ai lavori e degli utenti a comprendere le motivazioni dietro le decisioni di un modello è un aspetto preoccupante che dovrebbe destare l'attenzione della comunità scientifica. In effetti, tale problematica è stata affrontata sin dagli albori del Machine Learning, e si trovano vari articoli in letteratura, a partire dalla metà degli anni 80 [Ras et al., 2018], che cercano di porre una soluzione al problema (e.g. [Andrews et al., 1995]). Tuttavia, solo di recente la questione è diventata un aspetto critico su cui è necessario intervenire al più presto, proprio per l'ubiquità

ed il livello di complessità di tali sistemi.

Nell'introduzione a quest'elaborato abbiamo analizzato alcuni casi di successo nell'uso dei modelli di Machine Learning, ciò ci ha aiutato a comprendere il potenziale di queste tecnologie. Per capire la necessità di introdurre dei meccanismi atti ad aumentare l'interpretabilità dei sistemi in questione è utile osservare i casi di insuccesso che si sono verificati nel corso degli anni. Tra i tanti evidenziamo i seguenti:

- Il 2010 è stato l'anno del “Flash Crash”, 20 minuti di panico a Wall Street in cui un complesso sistema software per lo stock trading ha determinato perdite in borsa per un trilione di dollari³.
- Nel 2011 un'assistente vocale cui è stato chiesto “Call me an ambulance” ha iniziato a chiamare l'utente “an ambulance”⁴.
- Nel 2015 Google ha dovuto rivedere il proprio algoritmo di classificazione delle immagini dopo che quest'ultimo aveva iniziato a classificare le immagini di persone di colore come gorilla⁵.
- Nel 2015 un lavoratore della Volkswagen ha perso la vita a causa di un malfunzionamento di un robot, con il quale collaborava nella catena di montaggio. La compagnia si è difesa affermando che il malfunzionamento era dipeso da un errore umano⁶.
- Nel 2016 è avvenuto il primo incidente mortale causato da una Tesla in modalità di guida autonoma. A perdere la vita è stato il conducente dell'auto, distratto a guardare un film di Harry Potter. La compagnia ha sempre raccomandato agli automobilisti di rimanere vigili nonostante l'autopilot, proprio a causa dei limiti di quest'ultimo⁷.

³<https://www.theguardian.com/business/2015/apr/22/2010-flash-crash-new-york-stock-exchange-unfolded>

⁴<https://www.technologyreview.com/s/601897/tougher-turing-test-exposes-chatbots-stupidity/>

⁵https://www.huffingtonpost.com/2015/07/02/google-black-people-goril_n_7717008.html?guccounter=1

⁶<http://time.com/3944181/robot-kills-man-volkswagen-plant/>

⁷<https://www.theguardian.com/technology/2016/jul/01/tesla-driver-killed-autopilot-self-driving-car-harry-potter>

Dalla disamina di questi casi notevoli si evince che le misure prestazionali non possono più bastare per giudicare la bontà di un modello, vanno introdotte nuove metodologie che consentano di validare i suddetti sistemi. È opportuno, quindi, garantire che l'accuratezza di un modello sia derivante da una rappresentazione corretta del dominio su cui sta operando, facendo attenzione che i modelli non sfruttino degli artefatti presenti nei dati per raggiungere prestazioni elevate. Uno di questi concetti sulla cui base vanno validati i modelli è l'interpretabilità, nozione sulla quale ci soffermeremo in modo approfondito, cercando di impostare una discussione strutturata e coerente, in *Sezione 1.4*.

Le tecniche per l'interpretabilità dei sistemi di Machine Learning hanno anche altre applicazioni oltre a quelle relative la validazione di un modello. Una di queste è l'analisi dei dati scientifici. Come affermato in [Montavon et al., 2017], proprio grazie allo sviluppo di questo campo di ricerca è stato possibile effettuare scoperte di forte rilievo a partire dall'analisi di complessi sistemi fisici, chimici e biologici.

Ad esempio [Alipanahi et al., 2015] hanno utilizzato una tecnica innovativa per comprendere alcuni aspetti del genoma umano. Innanzitutto hanno addestrato una Convolutional Neural Network (CNN) [LeCun et al., 1999] a mappare una sequenza di DNA su delle regioni in cui si legano le proteine. Poi sfruttando la perturbation-based analysis sono riusciti a comprendere quali nucleotidi della sequenza fossero più rilevanti nello spiegare la correlazione con i siti di legame [Montavon et al., 2017]. Questo, ed altri esempi del genere sono mostrati in *Figura 1.3*.

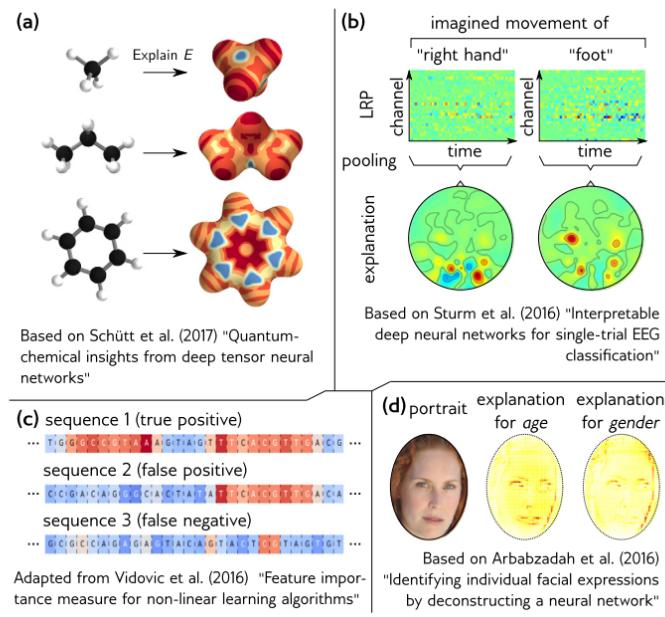


Figura 1.3: Immagine ottenuta da [Montavon et al., 2017]. Mostra l'uso di alcune tecniche di spiegazione di modelli complessi per l'analisi di dati scientifici

Un esempio di modello complesso: Deep Neural Network

Un esempio, cui spesso si fa riferimento per far comprendere la complessità dei modelli di Machine Learning attuali, è quello delle Deep Neural Network (DNN). Una rete neurale, in genere, è un modello matematico debolmente ispirato alla struttura del cervello umano. In particolare, esso si fonda sul concetto di neurone, l'unità di calcolo di base del cervello. Nel corso degli anni sono stati proposti vari approcci alla formalizzazione delle reti neurali. Al giorno d'oggi le varianti sono così numerose che diventa difficile anche tenerne traccia. Possiamo affermare che una delle formalizzazioni di successo è stata quella delle reti feed forward a strati full connected. Questo modello prevede la seguente organizzazione:

- La rete è organizzata a strati, o livelli
- Ogni strato è composto da un determinato numero di neuroni
- Ogni neurone di un livello prende in input tutti gli output dei neuroni dello strato precedente, ne fa una somma pesata, aggiunge una quantità detta bias e fornisce questo valore in input ad una funzione detta di attivazione che calcola l'output del neurone.

- I livelli intermedi, tra quello di input (il primo) e quello di output (l'ultimo), sono detti hidden layer.

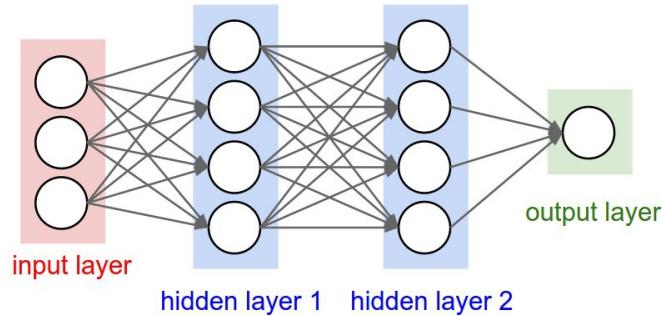


Figura 1.4: Modello di una rete neurale a strati full connected

Le Deep Neural Networks, nascono come estremizzazione del concetto di rete neurale, ma in generale ogni rete neurale con più di un hidden layer può essere considerata deep. Allo stato attuale le reti più complesse hanno anche svariate decine di livelli di profondità (di vario tipo, non solo full connected, e.g. livelli convoluzionali), un esempio lampante in tal senso è l'architettura di GoogLeNet uno dei modelli più complessi proposti da Google, che necessita di quasi 7 milioni di parametri.

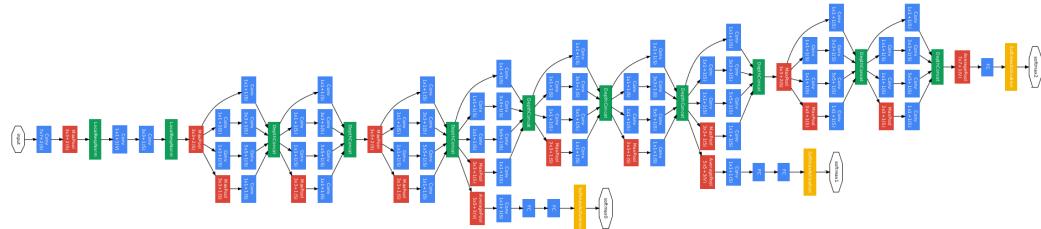


Figura 1.5: Architettura del modello GoogLeNet

Dunque, come si può comprendere dalla discussione appena effettuata, ripercorrere il processo di elaborazione di una rete con milioni di parametri è un procedimento complesso, che richiede tempo e non aumenta la conoscenza del sistema in modo significativo. Pertanto, è opportuno sviluppare delle tecniche che consentano di ricavare nuove informazioni su tali modelli senza la necessità di scrutarne il funzionamento interno.

1.3 Il ruolo della legge

Come accennato in precedenza, i sistemi di apprendimento automatico sono ormai presenti in molti settori, spesso critici (e.g. autonomous driving, medicine, financial markets, etc.). C'è quindi la possibilità che le scelte effettuate da tali modelli in produzione abbiano un impatto significativo sulla società. In altre parole, ci potrebbero essere delle conseguenze potenzialmente disastrose derivanti dall'utilizzo di tali sistemi.

Si pensi all'emblematico caso COMPAS: un software per predire il rischio di crimine recidivo, utilizzato nelle corti americane e venduto come black box dalla Northpointe. Un'indagine approfondita dei giornalisti di propublica.org ha evidenziato come tale software fosse estremamente discriminatorio nei confronti degli afroamericani. Per questi ultimi, infatti, il rischio di recidiva risultava sempre doppio rispetto ai detenuti bianchi⁸.

Dunque, diventa una necessità comprendere come regolamentare tali software in modo corretto. Infatti, come chiaramente affermato in [Doshi-Velez et al., 2017], “siamo in un momento storico per cui delle scelte giuste, in campo legislativo, sul quando richiedere una spiegazione potrebbero aiutare a prevenire scenari negativi per i sistemi di AI, mentre delle decisioni sbagliate non solo fallirebbero ad attribuire la giusta responsabilità ai sistemi di AI, ma arresterebbero lo sviluppo di tecnologie da cui l'uomo potrebbe trarre grande beneficio”.

In tal senso negli scorsi anni c'è stato un grande sforzo istituzionale per cercare di regolamentare l'utilizzo di sistemi in grado di prendere decisioni in modo autonomo. Il risultato di tale attività è il GDPR (General Data Protection Regulation) [Goodman and Flaxman, 2017] che andiamo ad analizzare in dettaglio nella sezione seguente.

1.3.1 GDPR - General Data Protection Regulation

Il GDPR, o General Data Protection Regulation[Goodman and Flaxman, 2017], è un documento mediante il quale il Parlamento Europeo ha proposto, nel 2016,

⁸<https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

un insieme di norme per regolamentare l'attività delle aziende che operano con dati appartenenti a cittadini della Comunità Europea.

Tra gli articoli più rilevanti vi sono, senza ombra di dubbio:

- **Articolo 17.** Anche conosciuto come *diritto all'oblio*. Sancisce il diritto di una persona a rimuovere ogni traccia della propria attività su internet.
- **Articolo 44.** Impone leggi molto severe per la tutela della privacy, e quindi per la gestione dei dati dei cittadini UE da parte di aziende straniere.
- **Articolo 22.** *“Automated individual decision-making, including profiling”*. Pone delle importanti limitazioni sull'utilizzo di software in grado di prendere decisioni in modo autonomo che possano influenzare gli utenti.

Proprio quest'ultimo risulta di particolare interesse per il presente elaborato, e pertanto ne forniamo la versione integrale nell'immagine 1.6.

Article 22. Automated individual decision making, including profiling	
1.	The data subject shall have the right not to be subject to a decision based solely on automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly affects him or her.
2.	Paragraph 1 shall not apply if the decision: (a) is necessary for entering into, or performance of, a contract between the data subject and a data controller; (b) is authorised by Union or Member State law to which the controller is subject and which also lays down suitable measures to safeguard the data subject's rights and freedoms and legitimate interests; or (c) is based on the data subject's explicit consent.
3.	In the cases referred to in points (a) and (c) of paragraph 2, the data controller shall implement suitable measures to safeguard the data subject's rights and freedoms and legitimate interests, at least the right to obtain human intervention on the part of the controller, to express his or her point of view and to contest the decision.
4.	Decisions referred to in paragraph 2 shall not be based on special categories of personal data referred to in Article 9(1), unless point (a) or (g) of Article 9(2) apply and suitable measures to safeguard the data subject's rights and freedoms and legitimate interests are in place.

Figura 1.6: GDPR - Articolo 22

Dall'analisi del suddetto articolo emerge sostanzialmente l'affermarsi di un nuovo **“diritto alla spiegazione”**. Un utente affetto da una decisione presa in modo autonomo da un algoritmo, può richiedere che gli venga fornita una spiegazione (elaboreremo nelle sezioni successive sul concetto di spiegazione) e può

intraprendere azioni contro di essa.

Questo diritto va, però, interpretato in modo corretto. Pensare a tale norma come ad una barriera per lo sviluppo di sistemi autonomi è un errore. È vero, essa impone restrizioni importanti sull'uso degli attuali modelli di Machine Learning, per i quali sarebbe necessaria una profonda ristrutturazione architettonale. Tuttavia, sarebbe più corretto inquadrare questa disposizione nel contesto dell'opportunità. Grazie a questo diritto, infatti, sarà possibile attrarre investimenti atti allo sviluppo di sottosistemi con lo scopo di inserire un elevato grado di **human interpretability** nel campo della progettazione algoritmica. Dunque, in definitiva, l'analisi di questo requisito pone la seguente domanda: “*Che significa e di cosa c'è bisogno per spiegare la decisione di un algoritmo?*” [Goodman and Flaxman, 2017].

Cercheremo di rispondere a tale domanda nelle prossime sezioni.

1.4 Interpretabilità

Per comprendere il bisogno di interpretabilità è utile porsi una domanda: “Se fossi ritenuto responsabile per la decisione presa da un macchina, in un contesto che abbia un forte impatto su un individuo in termini finanziari, di sicurezza, o di salute, ti fideresti ciecamente delle decisioni di tale macchina?” [Doran et al., 2017] La spiegazione, cui si è accennato nella sezione precedente, è solo uno dei mezzi per introdurre interpretabilità nel sistema, ma cos’è effettivamente l’interpretabilità? Nel corso degli anni sono stati proposti vari approcci all’analisi di questo concetto sfuggente, spesso usato in senso quasi matematico, come fosse un assioma [Lipton, 2016]. Di seguito cercheremo di riassumere le principali idee proposte e chiariremo la nostra posizione sull’argomento, in modo da inquadrare correttamente il contesto in cui è stata sviluppata la tecnica proposta in questo elaborato di tesi.

[Montavon et al., 2017] ritengono che un’interpretazione consista nel “*mappare un concetto astratto (e.g. una classe predetta) su un dominio interpretabile (i.e. un dominio che ha senso per l'uomo)*”. Esempi di domini interpretabili sono le immagini ed il testo. Nonostante sia un buon punto di partenza per iniziare a

ragionare sulla nozione di interpretabilità, questa definizione riesce a catturare solo una piccola porzione del concetto in questione. Quest’ultimo si va sempre più delineando come una nozione altamente sfaccettata la cui essenza risulta particolarmente difficile da formalizzare in una breve definizione.

Seguendo questa linea di pensiero [Doshi-Velez and Kim, 2017] ritengono che l’interpretabilità, nel campo dell’apprendimento automatico, sia la “*capacità di spiegare o presentare in termini comprensibili ad un essere umano*”. Le autrici specificano più volte nell’articolo che tale definizione va intesa come punto di partenza per iniziare a ragionare sul problema dell’interpretabilità. In particolare, viene evidenziato come l’interpretabilità vada utilizzata per confermare altre proprietà desiderabili in un sistema di ML. Tra queste caratteristiche troviamo sicuramente nozioni come:

- **Imparzialità.** Le minoranze non devono essere oggetto di discriminazione.
- **Privacy.** I modelli di ML devono rispettare e proteggere informazioni sensibili nei dati, evitando di usarle in modo improprio e/o di esporle a soggetti che non ne detengono la proprietà.
- **Affidabilità e Robustezza.** Gli algoritmi utilizzati, oltre ad essere performanti, devono essere robusti rispetto a modifiche dei parametri e/o dell’input.
- **Causalità.** Sebbene sia un aspetto ben lontano dall’essere incorporato completamente negli attuali modelli di ML⁹, questi ultimi dovrebbero almeno essere consistenti nel riflettere nell’output eventuali perturbazioni all’input.
- **Fiducia.** Gli utenti devono poter essere sicuri del fatto che il modello opererà in modo corretto su dati del mondo reale.

Un approccio analitico simile è utilizzato da [Lipton, 2016], che cerca di rendere il concetto in esame più formale. Oltre a condividere in gran parte la visione di [Doshi-Velez and Kim, 2017], ritenendo l’interpretabilità un’unione di idee

⁹<https://www.quantamagazine.org/to-build-truly-intelligent-machines-teach-them-cause-and-effect-20180515/>

piuttosto che un concetto monolitico, introduce una suddivisione della nozione presa in esame in due filoni principali, sulla base delle proprietà dei modelli interpretabili: trasparenza ed interpretabilità post-hoc.

1.4.1 Trasparenza

Nella sua accezione più semplice la trasparenza è l'esatto opposto della opacità o black-boxness. In tal senso, essa “valuta” la comprensione di un modello da parte di un utente. [Lipton, 2016] scinde questo concetto in 3 parti complementari:

- **Simulability.** Prende in considerazione la capacità di un utente di replicare, in un tempo ragionevole, le computazioni effettuate da un modello. In tal senso i modelli più semplici, come la regressione lineare, sembrano essere più interpretabili. Tuttavia questa visione non è del tutto veritiera, poiché anche modelli “semplici” applicati a dati con dimensionalità estremamente elevata non soddisfano questo requisito.
- **Decomposability.** È inerente alla comprensione da parte di un utente delle singole componenti di un modello (i.e. parametri, input, etc.). Anche in questo caso i modelli più semplici sembrano essere i più trasparenti. Bisogna però fare molta attenzione perché spesso i parametri che sembrano essere più comprensibili sono dipendenti da fattori che sono stati esclusi dal processo decisionale.
- **Algorithmic transparency.** Si riferisce alla complessità degli algoritmi utilizzati per la gestione del modello (i.e. training, inferenza, etc.).

[Weller, 2017] critica fortemente l’uso incondizionato della trasparenza, sostenendo che essa potrebbe causare vari danni e diminuire l’imparzialità del sistema. Infatti, esperimenti portati avanti da [Hammond and Axelrod, 2006] hanno mostrato come in alcuni contesti un aumento di trasparenza possa portare a risvolti davvero poco auspicabili, in cui l’imparzialità viene quasi del tutto azzerata. Oltre a ciò una maggiore trasparenza è spesso in contrasto con una necessità di privacy da parte degli utenti.

1.4.2 Interpretabilità post-hoc

L’interpretabilità post-hoc si differenzia dalla trasparenza nel modo in cui cerca di estrarre informazioni utili da un modello. Essa non cerca di capire il funzionamento del modello esplorandone le meccaniche interne, bensì considera quest’ultimo come una black-box sulla cui impalcatura costruire un sistema esterno che tenti di estrapolare la “conoscenza” presente nel modello stesso [Montavon et al., 2017]. Un vantaggio delle tecniche che prendono in considerazione questo tipo di interpretabilità, sta nel fatto che esse possono operare su modelli già addestrati e pertanto possono essere integrate in sistemi già esistenti senza intaccarne le prestazioni. Per questo motivo, spesso in letteratura, ci si riferisce alle tecniche che tentano di chiarire il funzionamento di un modello ottimizzando l’interpretabilità post-hoc come “*model-agnostic techniques*”. Bisogna fare attenzione, quando si creano tecniche il cui scopo è aumentare l’interpretabilità post-hoc, a non introdurre all’interno di tali sistemi dei difetti tipici dell’essere umano [Lipton, 2016]. In tal caso gli algoritmi potrebbero tendere a presentare spiegazioni plausibili, ma decisamente fuorvianti rispetto agli obiettivi posti dalla necessità di interpretabilità. Infatti, è stato mostrato più volte come gli esseri umani tendano a fornire giustificazioni all’apparenza attendibili, che celano tuttavia motivazioni oscure. In tal senso, vari giornalisti hanno evidenziato come, spesso, le ammissioni in grandi università o le assunzioni in azienda, giustificate sulla base di fattori come la leadership o l’originalità, siano in realtà fondate su discriminazioni a sfondo sessuale o razzista¹⁰.

Un’ulteriore rifinitura presente in letteratura, prende in considerazione il livello di granularità dell’interpretabilità. Essa scinde questo concetto in “*globale*” e “*locale*”. Per interpretabilità globale si intende lo sforzo di comprensione di un intero modello, mentre con interpretabilità locale ci si riferisce alla comprensione del perché un modello ha avuto un determinato comportamento su un’istanza specifica.

¹⁰https://www.nytimes.com/2014/11/25/opinion/is-harvard-unfair-to-asian-americans.html?_r=0 Consultato in data 5/10/2018.

In definitiva riteniamo che il concetto di interpretabilità vada inteso come un set di sotto-nozioni che non è detto possano essere ottimizzate tutte contemporaneamente (e.g. trasparenza vs privacy). In tal senso, il contesto in cui si opererà, detterà la linea d’azione da tenere, e quindi le componenti più rilevanti da introdurre nel sistema. Per comprendere meglio questa visione del concetto di interpretabilità, si pensi ad un dispositivo di uso comune, ad esempio un aspirapolvere, in cui viene inserita una smart feature: aumentare l’interpretabilità in tale strumento non si riflette nell’introduzione di una maggiore l’imparzialità nel sistema, bensì nell’aggiunta di sicurezza.

Sebbene varie tecniche saranno prese in esame nel corso di questo elaborato, la soluzione proposta al problema dell’interpretabilità (Capitolo 3) sarà inquadrata nel contesto della interpretabilità post-hoc.

1.4.3 Valutare i metodi per l’interpretabilità

[Doshi-Velez and Kim, 2017] propongono la seguente tassonomia per la valutazione dei metodi utilizzati al fine di introdurre un maggior grado di interpretabilità nel sistema:

- ***Application-grounded.*** Consiste nell’effettuare esperimenti per un’applicazione specifica, con supporto degli esperti del settore. Questa strategia prende spunto dalle tecniche di testing nel contesto dell’HCI. È un approccio molto costoso, ma estremamente efficace.
- ***Human-grounded.*** Questa metodologia fa uso di persone non esperte del domino e pertanto consente di valutare aspetti più generali della soluzione introdotta per risolvere il problema dell’interpretability. Ha un costo minore, ma non consente il testing di caratteristiche specifiche per l’applicazione.
- ***Functionally-grounded.*** Non richiedono esperimenti con persone. Utilizzano una definizione formale di interpretabilità per effettuare una valutazione. Hanno un costo molto basso, ma la loro efficacia dipende dal proxy

utilizzato e non è ancora chiaro quale sia il criterio giusto per la scelta del proxy (e.g. un proxy potrebbe essere un albero decisionale che approssima una rete neurale).

1.5 Spiegazione

Abbiamo affermato che la spiegazione può essere utilizzata come un mezzo per migliorare l'interpretabilità del sistema; ma cos'è effettivamente una spiegazione, quali sono le caratteristiche costitutive della stessa e quando sorge la necessità di produrne una? Le spiegazioni sono uno strumento fondamentale al nostro senso di comprensione (esse rispondono alla domanda “Perché?”) e pervadono così tanto la nostra quotidianità che spesso non ci rendiamo di quanto sia intenso il loro utilizzo.

Nonostante siano onnipresenti, una definizione formale del concetto di spiegazione rimane sfuggente. Nella sua accezione più generica, l'atto di spiegare vuol dire *“Far capire, chiarire, rendere chiaro e intelligibile qualcosa di oscuro e di difficile comprensione”*¹¹. In tal senso, [Lombrozo, 2006] riprende questo concetto affermando che le spiegazioni sono *“la valuta con la quale ci scambiamo la conoscenza”*. L'autrice ritiene, inoltre, che le spiegazioni siano in grado di *“validare nuove informazioni sulla base delle conoscenze pregresse, e lo facciano in un modo tale da promuovere la generalità”*. Attenzione però, secondo questa visione, siccome esse si basano su conoscenza pregressa, sono soggette alla veridicità della stessa. Quindi, se si dovesse generare una spiegazione basata su false credenze, avremmo un perpetuarsi dell'inaccuratezza anche nelle nuove informazioni.

Un approccio più critico alla spiegazione è adottato da [Wilson and Keil, 1998], i quali ritengono che *“la spiegazione di un dato fenomeno sia un tentativo, apparentemente di successo, di incrementare la comprensione di quel fenomeno”*. Questa affermazione impone delle proprietà ben precise ad una spiegazione, tra cui evidenziamo:

¹¹Treccani, consultata in data 26/07/2018 - [http://www.treccani.it/vocabolario/
spiegare](http://www.treccani.it/vocabolario/spiegare)

- L'efficacia di una spiegazione dipende dal modo in cui viene comunicata.
- Siccome la spiegazione è un tentativo, potrebbe fallire.
- Dato che ogni fenomeno è inevitabilmente collegato ad altri, la spiegazione del suddetto evento può portare come side-effect anche una comprensione maggiore di quelli correlati.

Gli autori inoltre aggiungono che la struttura stessa di una spiegazione può variare fortemente a seconda di cosa si sta chiarendo. Infatti una spiegazione del perché si è arrivati tardi a lavoro, condivide poco o nulla con una dimostrazione di un teorema da parte di un matematico.

Entrando più nello specifico, nel campo del Machine Learning, [Ras et al., 2018] ritengono che una spiegazione dovrebbe, idealmente, esibire le seguenti proprietà:

- **Fedeltà.** La spiegazione dovrebbe essere fedele alle caratteristiche del modello preso in considerazione.
- **Chiarezza.** La spiegazione non dovrebbe essere ambigua.
- **Parsimonia.** La spiegazione dovrebbe essere concisa e considerare solo pochi elementi fondamentali. Il principio del Rasoio di Occam (i.e. “*pluritas non est ponenda sine necessitate ponendi*”) è alla base di questa caratteristica.
- **Generalità.** La spiegazione dovrebbe essere in grado di spiegare un gran numero di modelli. Idealmente dovrebbe essere model-agnostic.
- **Potere espressivo.** La spiegazione dovrebbe essere in grado di delucidare un gran numero di fenomeni.

Aggiungiamo che una spiegazione dovrebbe essere anche in grado di adattare la propria struttura all'utente cui è rivolta. Inoltre, quest'ultimo può avere esigenze più o meno stringenti riguardo il tempo necessario alla comprensione della spiegazione.

1.5.1 Quando generare una spiegazione

Produrre una spiegazione ha un costo. Spesso si tende a dimenticare questo aspetto, quando in realtà esso ricopre un ruolo chiave nel comprendere se una spiegazione sia necessaria o meno.

In generale, [Doshi-Velez et al., 2017] suggeriscono che la generazione di una spiegazione ha senso se il risultato ottenuto dal processo di decision making è in conflitto con quello atteso. In questo caso il ruolo della spiegazione assume particolare rilievo poiché consente di riallineare la percezione che un utente ha del sistema con il suo funzionamento effettivo. Inoltre, la produzione di una spiegazione non può prescindere dall'impatto che ha avuto il risultato del processo decisionale sull'utente. Un esempio classico in tal senso è quello dell'investitore: egli non ha bisogno di fornire una spiegazione a se stesso in seguito ad un investimento sbagliato dei propri fondi; ha però l'obbligo di fornire tale spiegazione ad un cliente nel caso in cui i fondi investiti afferiscano a quest'ultimo. Infine, la produzione di una spiegazione dovrebbe, idealmente, consentire a chi ha subito il danno di intraprendere un'azione legale atta a ricevere almeno un indennizzo economico.

Ovviamente i principi appena elencati non sono esaustivi e sono soggetti a cambiamenti nel tempo in base all'evoluzione che si avrà nel mondo del Machine Learning. Ci sono circostanze in cui, come visto, non solo produrre una spiegazione non ha alcuna conseguenza positiva, ma risulta addirittura un peso (economico e temporale) nello sviluppo di sistemi più affidabili. In particolare, gli stessi autori individuano due casi in cui è possibile evitare di fornire una spiegazione:

- **Evidenza teorica.** È la forma di “accountability” cui si dovrebbe aspirare, soprattutto nel lungo termine. Ci sono situazioni in cui è possibile mostrare una prova teorica del funzionamento corretto del sistema. Si pensi ad esempio all'uso della crittografia nei sistemi di messaggistica istantanea, in tal caso le basi teoriche giustificano l'utilizzo del sistema in ambienti reali e consentono di evitare la generazione di spiegazioni.
- **Evidenza Empirica (o Statistica).** È una forma di evidenza più de-

bole, poiché soggetta al giudizio di chi la valuta. Essa giustifica, in modo implicito, l'uso del sistema, sulla base dei risultati empirici collezionati. Se ne auspica l'utilizzo in casi di discriminazione, contesto in cui l'evidenza statistica può mettere in luce aspetti del sistema che non erano stati presi in considerazione.

1.6 Direzione in cui si sviluppa l'elaborato

Per riassumere, in questo elaborato ci soffermeremo sul problema dell'interpretabilità nel Machine Learning, considerando esclusivamente problematiche relative al supervised learning, senza soffermarci su altre metodologie come il reinforcement learning o l'interactive learning. In particolare ci focalizzeremo sullo studio della classificazione delle immagini e delle principali problematiche di questo campo (per una discussione approfondita di tale argomento fare riferimento al paragrafo: “*Problemi di Classificazione*”). Tenteremo, pertanto, di rispondere a domande del tipo:

“Perché quest’immagine è stata classificata dal modello come una scarpa al 99%?”

Sebbene gli esempi che porteremo avanti potrebbero, a prima vista, sembrare eccessivamente semplici, essi saranno sempre tesi a far comprendere le potenzialità della tecnica se propriamente applicata ad un ambiente di produzione. Cercheremo, inoltre, di astrarci il più possibile da una struttura specifica di un modello così da concentrarci sugli aspetti concettuali che sono alla base dei processi decisionali nell’ambito dell’apprendimento automatico. Infatti, così come in [Pedreschi et al., 2018], riteniamo che i framework per le spiegazioni di modelli black box debbano essere:

- **model-agnostic**, in modo che possano essere modulari e quindi applicabili ad una vasta gamma di modelli già addestrati.
- **basati su un ragionamento logico**, così che le spiegazioni possano risultare comprensibili a diversi tipi di utenti (esperti e non).

- **locali e globali**, in modo da spiegare sia l’inferenza su singoli elementi che il comportamento del modello nel suo complesso.
- **altamente fedeli**, per fornire un’approssimazione quanto più possibile accurata dei modelli black-box.

In definitiva, per inquadrare bene il lavoro che andiamo a presentare nei capitoli successivi, è utile pensare alla tecnica che andiamo a sviluppare come ad uno dei tanti strumenti di un coltellino svizzero. In tal senso tutti gli strumenti consentono di affrontare il problema dell’interpretabilità, ognuno da un punto di vista diverso ed ugualmente importante al raggiungimento dell’obiettivo finale. Non stiamo cercando quindi di risolvere tale questione con la creazione di un metodo risolutivo onnicomprensivo, bensì mettiamo a disposizione della comunità un nuovo punto di partenza sul quale ragionare per cercare di affrontare la suddetta problematica.

Problemi di Classificazione

[Wasserman, 2010] definisce formalmente la classificazione come il problema di predire il valore di una variabile aleatoria discreta Y sulla base del valore di un’altra variabile aleatoria X . In particolare, consideriamo delle variabili indipendenti e identicamente distribuite $(X_1, Y_1), \dots, (X_n, Y_n)$ dove

$$X_i = (X_{i1}, \dots, X_{id}) \in \chi \subset \mathbb{R}^d$$

è un vettore d-dimensionale e Y_i assume valori in un insieme finito γ . Una regola di classificazione è una funzione $h : \chi \rightarrow \gamma$. Sulla base di questa regola, quando si osserva un nuovo elemento X si predice $Y = h(X)$.

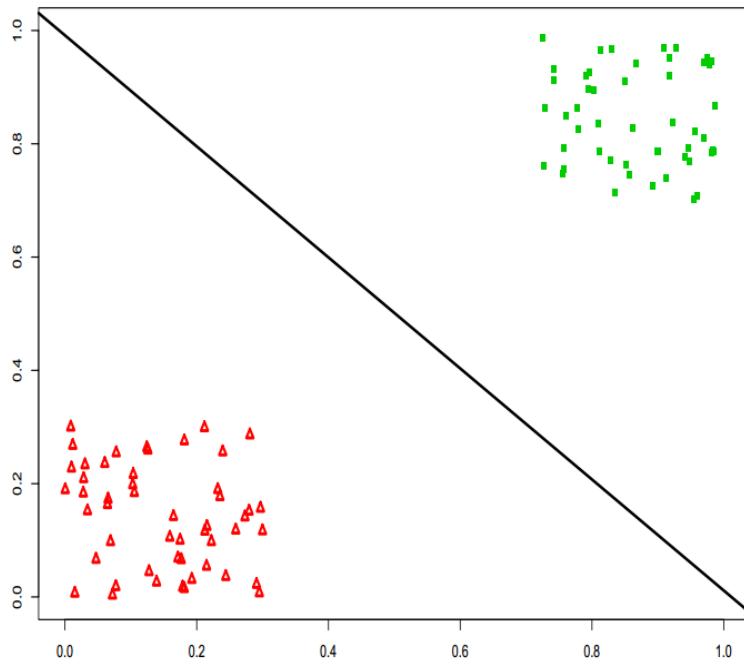


Figura 1.7: Immagine tratta da [Wasserman, 2010]. Mostra un classificatore lineare su dati rappresentati dalle due variabili x_1 e x_2

Per comprendere come tale concetto si applica al campo dell’elaborazione delle immagini possiamo immaginare la seguente situazione, tratta da [Bishop, 1995]. Prendiamo in considerazione il problema di dover distinguere due immagini di caratteri scritti a mano ’a’ e ’b’. Vogliamo individuare un algoritmo che riesca a distinguere i due caratteri. Un’immagine è rappresentata come un vettore x di pixel x_i , ognuno dei quali assume un valore tra 0 ed 1 (ci poniamo nel caso delle immagini in scala di grigio). Per cui diremo che:

$$x = (x_1, \dots, x_d)$$

Lo scopo del problema di classificazione in questione è quello di sviluppare un algoritmo in grado di assegnare ad ogni immagine x ad una classe C_k con $k = 1, 2$, in modo che $C_1 = a$ e $C_2 = b$. Per ottenere la soluzione a questo problema supponiamo di avere a disposizione un insieme, molto grande, di immagini già classificate da un essere umano (i.e. assegnate alla classe C_1 o C_2). Sulla base di questo *data set* è possibile, apprendere un classificatore (o *decision boundary*) ossia una regola che consenta di mappare le immagini sulle classi di appartenenza,

con un certo errore di classificazione. La procedura necessaria all'apprendimento di tale regola varia a seconda del classificatore scelto e non sarà approfondita in questo elaborato.

Capitolo 2

Lo stato dell'arte

Di recente, c'è stato un notevole incremento delle attività di ricerca nell'ambito della Machine Learning Interpretability. Basti pensare che solo nell'intervallo di anni tra il 2015 ed il 2017 sono stati pubblicati più di 20.000 articoli inerenti il suddetto campo e tale numero è in continuo aumento¹. Di conseguenza, è stato sviluppato un gran numero di tecniche atte a risolvere vari aspetti dell'interpretabilità.

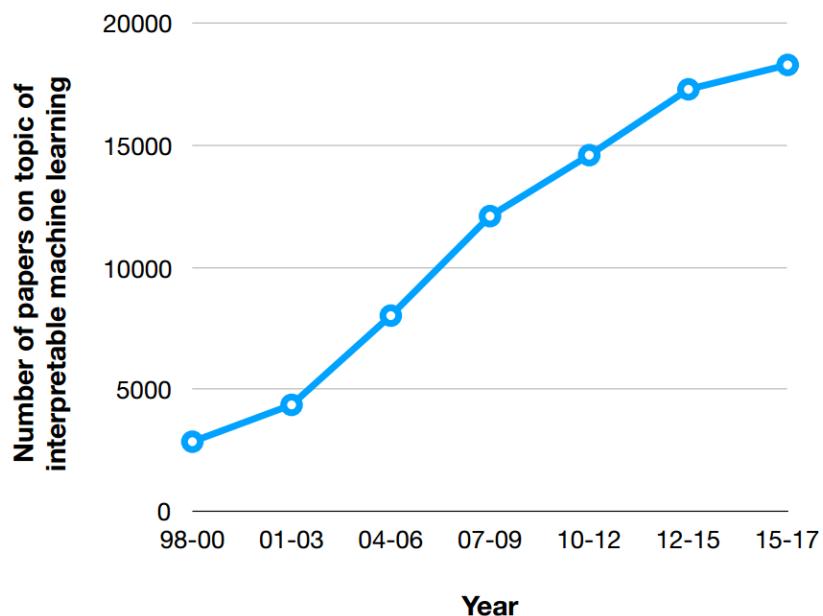


Figura 2.1: Numero di paper pubblicati per anno riguardanti la MLI¹

¹https://beenkim.github.io/papers/BeenK_FinaleDV_ICML2017_tutorial.pdf

Dunque, tenere traccia di tutte le soluzioni generate, oltre a non essere l’obiettivo del presente lavoro, non è un compito ragionevole. Nel seguente capitolo, quindi, esamineremo una tassonomia che cerca di standardizzare la categorizzazione delle tecniche nell’ambito della Machine Learning Interpretability. In seguito, approfondiremo i vari approcci presenti in letteratura, sempre inerenti la tecnica che intendiamo sviluppare. Ne analizzeremo i punti di forza e di debolezza, proponendo un percorso che porterà il lettore a comprendere l’idea alla base della soluzione proposta nel presente elaborato.

2.1 Una tassonomia condivisa

Negli ultimi anni sono stati fatti vari sforzi per introdurre una tassonomia standard e comunemente accettata dai ricercatori che operano nell’ambito della Machine Learning interpretability. In particolare un primo tentativo è stato portato a termine da [Montavon et al., 2017], i quali hanno il merito di aver delineato la strada da intraprendere, impostando tuttavia un discorso poco esaustivo. Un approccio leggermente differente è stato adottato da [Ras et al., 2018] che hanno classificato le varie tecniche in modo più formale e hanno posto le basi per una tassonomia più completa. Sebbene su questi due lavori si baserà gran parte del contenuto di questo capitolo, è sempre bene tenere a mente che essi vanno intesi come spunti sui quali iniziare a ragionare in modo strutturato e costruttivo su questi temi. Questo soprattutto perché, essendo l’interpretabilità nel Machine Learning un filone di ricerca relativamente nuovo, risulta in continua evoluzione e con il tempo potrebbero emergere nuove necessità e quindi nuove strategie risolutive che non è possibile prevedere allo stato attuale delle cose.

Ciò detto, [Ras et al., 2018] propongono la seguente categorizzazione dei metodi: **rule-extraction methods**, **attribution methods**, **intrinsic methods**.

Rule-extraction methods Basati sull’estrazione di “regole” che approssimano il processo decisionale di un modello complesso, utilizzando solo l’input e l’output dello stesso. La rule extraction può avvenire sia a livello *locale* (spiegazione di

una singola istanza su cui è stata fatta inferenza) che *globale* (spiegazione del modello nel suo complesso). La prima modalità consente di estrarre regole più fedeli al vero comportamento del modello. Questa categoria di metodi è ritenuta una delle più efficaci e viene ulteriormente scissa in:

- **Pedagogical approach.** È uno degli approcci di maggiore successo, a questa categoria afferisce il metodo LIME [Ribeiro et al., 2016], presentato nella *Sezione 2.2.3*. Lo scopo è quello di intendere l'estrazione di regole come un task di apprendimento in cui il target è il modello e le feature di input sono semplicemente quelle in input al modello stesso. Consente di generare sistemi di spiegazione model-agnostic (il modello è inteso come una black-box cui non è possibile accedere). Questa caratteristica rende tali metodi ampiamente adottati poiché adattabili anche a modelli sviluppati in passato, altamente prestanti, per i quali risulta impossibile una modifica strutturale.
- **Eclectic approach.** Le tecniche che fanno parte di questa categoria utilizzano la conoscenza della struttura interna del modello per creare dei sistemi di apprendimento simbolico. Sono sicuramente tra le tecniche meno utilizzate, ma garantiscono un formalismo più rigido.
- **Decompositional approach.** A questa categoria fanno riferimento tutte quelle tecniche che scompongono una Deep Neural Network (DNN), o qualsiasi modello complesso, in sotto-sistemi più piccoli. Per fare ciò è necessario avere una conoscenza della struttura interna del modello e dei suoi input/output. Pertanto, queste tecniche, così come quelle Eclectic, non sono model-agnostic.

Questa ulteriore classificazione è stata proposta da [Andrews et al., 1995] e mostra come la necessità di interpretabilità sia stata forte sin dai primi anni della ricerca su reti neurali complesse.

Attribution methods In alcuni casi conosciuti anche come visualization methods. Consentono di risolvere aspetti dell'interpretabilità legati al processing

delle immagini e del testo. Operano apportando un cambiamento all'input o alle componenti interne al modello e registrando come le modifiche impattano le prestazioni del sistema. In alternativa, assegnano uno score di rilevanza alle feature dell'input che influenzano maggiormente l'output del modello. La maggior parte delle metodologie che fanno parte di questa categoria sono model-agnostic ed esibiscono caratteristiche che le rendono estremamente intuitive. Proprio per questo, sono ampiamente adottate ed accettate nella comunità del Machine Learning.

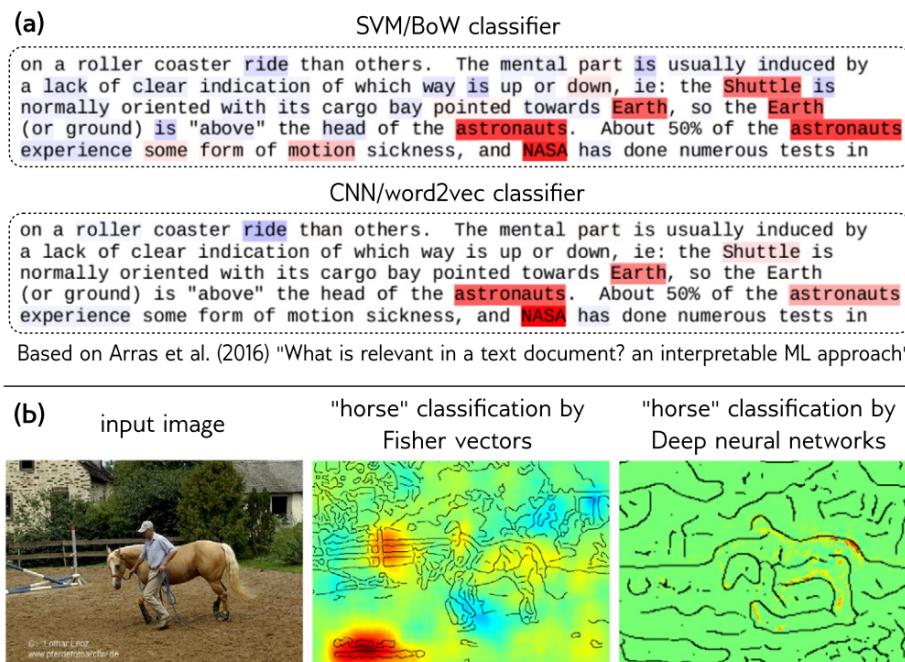


Figura 2.2: Esempi di attribution methods, ottenuti da [Montavon et al., 2017]

La loro popolarità tuttavia è spesso legata a forti critiche riguardanti la mancanza di affidabilità di questi metodi quando la spiegazione è sensibile a fattori che non contribuiscono alla predizione del modello [Kindermans et al., 2017a]. Sostanzialmente, è possibile forzare tali metodi a produrre una spiegazione che all'apparenza sembrerebbe ottima (secondo i parametri indicati nel Capitolo 1), ma in realtà nasconde degli artefatti che non hanno alcun nesso con le prestazioni del modello. Un caso eclatante in tal senso è mostrato nella *Sezione 2.2.1*. Inoltre, l'intuitività di questi metodi è allo stesso tempo un punto di forza e debolezza. A differenza dei rule extraction methods, i visualization methods sono soggetti ad interpretazione. Essi infatti forniscono una spiegazione visiva, che mostra

all’utente quali fattori dell’input hanno maggiore rilevanza nel processo decisionale che si riflette nell’output del modello. A questa categoria appartengono la maggior parte dei metodi presi in considerazione in questo capitolo.

Intrinsic methods Il cui scopo è quello di migliorare l’interpretabilità delle rappresentazioni interne al modello, con metodi che sono già parte dell’architettura del sistema. A differenza dei metodi precedenti, quelli che attengono a questa categoria non operano su modelli già addestrati, bensì cercano di aumentare l’interpretabilità di un modello modificandone la struttura, la loss function, la procedura di apprendimento e altre caratteristiche insite al modello stesso. Ovviamente operando sulla struttura stessa del sistema non possono essere model-agnostic, ma sono comunque di fondamentale importanza per lo sviluppo di modelli migliori e pongono le basi per l’implementazione di attribution methods superiori in termini di fedeltà, chiarezza e parsimonia, rispetto a quelli attuali. Essendo dipendenti dalle singole architetture, i metodi di questa categoria non saranno affrontati nella discussione seguente.

2.2 Interpretabilità globale e locale

Come accennato in precedenza, per **interpretabilità globale** intendiamo lo sforzo di comprensione di un intero modello. Quindi la domanda che ci si pone in tale contesto è “Come si comporta il modello nella sua totalità?”. Una spiegazione del modello nel suo complesso può aiutarci a comprendere meglio la funzione di distribuzione che ha appreso, e la correlazione che intercorre tra le variabili di input e di output, ma le explanation prodotte saranno sempre un’approssimazione del funzionamento reale del modello stesso. Nel campo delle immagini uno degli strumenti principali per l’interpretabilità globale è la generazione di prototipi. Questa avviene mediante tecniche che cadono sotto il nome di feature visualization, atte a rispondere alla domanda precedente tramite la generazione di prototipi. Un prototipo è, nella sua accezione più generica, “*Con uso iperb., non com., chi presenta caratteristiche, qualità, difetti tipici di una determinata*

*categoria di persone in grado tale da risultare particolarmente rappresentativo di quella categoria*². Pertanto, nel campo della classificazione delle immagini, è possibile intendere un prototipo come un’immagine che esibisce le caratteristiche fondamentali della categoria cui appartiene. Mediante la generazione di tale immagine è possibile, in linea teorica, comprendere quali sono le caratteristiche che il modello ritiene siano fondamentali per una determinata categoria. Questo tipo di analisi porta ad una conoscenza più profonda del modello e del dataset utilizzato per l’apprendimento, consentendo così di attuare strategie per l’ottimizzazione di entrambi. Nella *Sezione 2.2.1* discuteremo in modo approfondito della generazione di prototipi, nel campo della classificazione delle immagini, mediante un framework noto come Activation Maximization. Ne analizzeremo i pro ed i contro, cercando di capire quali aspetti della tecnica possono essere utili per lo sviluppo della nostra metodologia.

L’**interpretabilità locale**, a differenza di quella globale, consente di comprendere il comportamento del modello in merito ad un’istanza specifica. Essa risponde alla domanda ”Perché il modello ha avuto un determinato comportamento nei confronti dell’esempio x ?”. Nel campo della classificazione delle immagini tale domanda può essere intesa nel seguente modo: “Quali sono le caratteristiche dell’esempio x che ne hanno determinato la classificazione nella categoria c ?”. Essendo riferite a specifiche istanze le spiegazioni prodotte nel contesto dell’interpretabilità locale sono molto più accurate di quelle globali e non soffrono di tutta una serie di problematiche legate alla generazione dei prototipi (che saranno analizzate in *Sezione 2.2.1*).

Dunque, nelle prossime sezioni: studieremo una tecnica per l’interpretabilità globale e capiremo com’è possibile adattarla alla generazione di spiegazioni per la local interpretability (*Sezione 2.2.1*), analizzeremo tecniche specifiche per quest’ultima (*Sezione 2.2.2*) e metodologie che sfruttano l’interpretabilità locale per fornire una comprensione globale del modello(*Sezione 2.2.3*).

²Treccani, consultata in data 21/09/2018 - <http://www.treccani.it/vocabolario/prototipo>

2.2.1 Activation Maximization (AM)

È un framework di ottimizzazione generico, che non nasce nel mondo della machine learning interpretability, ma si trova in letteratura sotto vari nomi. Per la prima volta compare nel contesto dell'interpretabilità di modelli complessi con [Erhan et al., 2009], i quali ne analizzano le potenzialità in riferimento alle reti neurali. Nel lavoro di Erhan e colleghi è stato usato per individuare dei pattern di input che massimizzano la risposta (activation) di un neurone in uno strato interno alla rete.

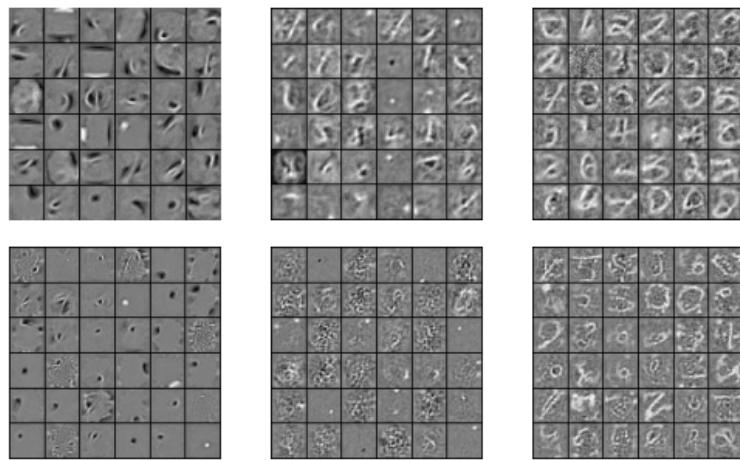


Figura 2.3: Immagine tratta da [Erhan et al., 2009] AM applicato a MNIST. Visualizzazione di 36 unità dal primo, secondo e terzo strato interno di una Deep Belief Network.

Nonostante ciò, il meccanismo che andiamo a descrivere, è applicabile a tutti i nodi di una rete ed in generale risulta agnostico rispetto al modello. Gli autori descrivono formalmente il framework come segue:

$$x^* = \max_{x \text{ s.t. } \|x\|=\rho} h_{ij}(\theta, x) \quad (2.1)$$

Con θ che denota l'insieme dei pesi e dei bias della rete, x l'immagine in input al modello, e h_{ij} l'attivazione di un'unità i in un livello j .

Il problema che AM tenta di risolvere è un problema di ottimizzazione non-convesso. Per questo motivo la risoluzione dello stesso avviene mediante ascesa del gradiente, che consente quantomeno di giungere ad un massimo locale, facil-

mente calcolabile in una rete neurale grazie alla tecnica della backpropagation. Siccome il nostro obiettivo finale è la creazione di una tecnica model-agnostic, dobbiamo pensare al modello come ad una black-box e quindi non ci interessa analizzare la risposta dei nodi interni di una rete ad un determinato input. Piuttosto, verifichiamo la capacità del framework di essere usato per la generazione di prototipi. Per descrivere il problema in questione utilizziamo la notazione proposta da [Simonyan et al., 2013] e ripresa da [Montavon et al., 2017]:

$$\max_x \log p(\omega_c|x) - \lambda \|x\|^2 \quad (2.2)$$

Dove $p(\omega_c|x)$ rappresenta la probabilità che l'input x appartenga alla classe ω_c , mentre $\|x\|^2$ è la norma L-2, un termine di regolarizzazione che implementa una preferenza per gli input che sono vicini all'origine. In questo modo è possibile individuare un pattern di input (i.e. un'immagine nel nostro caso) che rende la risposta del modello massima, rispetto ad una quantità di interesse. Dunque, come si genera, in termini pratici, un prototipo?

1. Si genera un'immagine con del rumore
2. Si fornisce tale immagine in input al modello
3. Si calcola la risposta del modello rispetto alla classe per cui si sta ottimizzando (i.e. $p(\omega_c|x)$)
4. Si calcola la quantità di interesse
5. Si modifica l'input (ossia l'immagine) in modo da massimizzare la quantità di interesse. Questo step può essere implementato in vari modi. Nelle reti neurali, si può utilizzare l'ascesa del gradiente (tenendo fissi pesi e bias e modificando solo l'input).

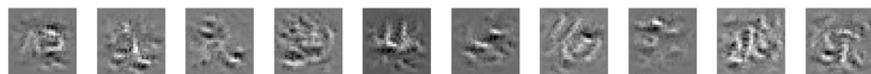


Figura 2.4: Prototipi per MNIST da 0 (a sinistra) a 9 (a destra), generati mediante AM, su Lenet-5 [Lecun et al., 1998], con la quantità di interesse proposta in 3.1

Idealmente, la generazione di un prototipo dovrebbe essere un task relativamente semplice. In realtà, riuscire a far funzionare la metodologia appena descritta non è per nulla banale. Come chiaramente descritto in [Olah et al., 2017], la riuscita della tecnica dipende fortemente dalla regolarizzazione utilizzata nella quantità di interesse. Infatti, senza alcuna regolarizzazione il framework produce delle immagini costituite essenzialmente da rumore e da pattern senza senso cui il modello risponde fortemente. Questa situazione è strettamente collegata al fenomeno degli adversarial examples [Szegedy et al., 2013], immagini che ingannano il modello ottenendo delle risposte completamente falsate. Capire come far funzionare correttamente la tecnica e che tipo di regolarizzazione utilizzare, è stato uno dei filoni di ricerca più attivi degli ultimi anni nel campo della feature visualization. La *Figura 2.5* mostra i vari stratagemmi utilizzati nel corso degli anni per migliorare il framework AM.

	Weak Regularization avoids misleading correlations, but is less connected to real use.	Strong Regularization gives more realistic examples at risk of misleading correlations.			
	Unregularized	Frequency Penalization	Transformation Robustness	Learned Prior	Dataset Examples
	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Erhan, et al., 2009 [3] Introduced core idea. Minimal regularization.					
	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Szegedy, et al., 2013 [11] Adversarial examples. Visualizes with dataset examples.					
	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Mahendran & Vedaldi, 2015 [7] Introduces total variation regularizer. Reconstructs input from representation.					
	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Nguyen, et al., 2015 [14] Explores counterexamples. Introduces image blurring.					
	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Mordvintsev, et al., 2015 [4] Introduced jitter & multi-scale. Explored GMM priors for classes.					
	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Øygard, et al., 2015 [15] Introduces gradient blurring. (Also uses jitter.)					
	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Tyka, et al., 2016 [16] Regularizes with bilateral filters. (Also uses jitter.)					
	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Mordvintsev, et al., 2016 [17] Normalizes gradient frequencies. (Also uses jitter.)					
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Nguyen, et al., 2016 [18] Parametrizes images with GAN generator.					
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Nguyen, et al., 2016 [19] Uses denoising autoencoder prior to make a generative model.					

Figura 2.5: Iterazioni di AM nel corso degli anni, immagine ottenuta da [Olah et al., 2017]

Le tre colonne centrali della tabella in *Figura 2.5* mostrano che in letteratura sono state usate principalmente tre famiglie di regolarizzazione:

- **Frequency penalization.** Ha un impatto diretto sul rumore e sui pattern ad alta frequenza. Principalmente vengono usate la *total variation*, che penalizza la varianza tra pixel vicini, e i *bilateral filter*, filtri di blur che conservano gli edge applicati all’immagine ad ogni iterazione dell’algoritmo.
- **Transformation robustness.** Si cercano degli input che ottengono una risposta molto forte dal modello anche in caso di piccole trasformazioni degli stessi. Praticamente ad ogni passo dell’algoritmo ti applicano delle trasformazioni all’immagine (e.g. rotazione, scala, jitter, ecc.)
- **Learned priors.** Si apprendono dei modelli generativi sul dataset in esame, ad esempio GAN (Generative Adversarial Network) o VAE (Variational Autoencoder), che mappano punti di uno spazio latente su esempi del dataset. Invece di ottimizzare sull’immagine si ottimizza poi sullo spazio latente. Questo approccio consente di ottenere immagini ad alta risoluzione.

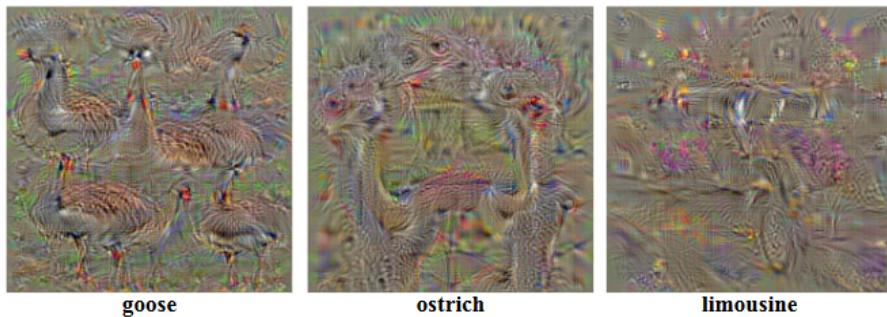


Figura 2.6: Esempio di AM con regolarizzazione $\lambda||x||^2$, su ConvNet, addestrato su ILSVRC-2013. Immagine tratta da [Simonyan et al., 2013]

Tra le varie versioni presenti in letteratura, quella di maggiore successo in termini di generazione di un’immagine sintetica simile ad una reale è proposta da [Nguyen et al., 2016a]. Per ottenere i risultati mostrati in *Figura 2.7* gli autori integrano un modello generativo con AM. Nello specifico essi fanno uso delle GAN [Goodfellow et al., 2014]. Dal momento in cui questa variante dell’Activation Maximization non è stata sviluppata al fine di aumentare l’interpretabilità

globale di un modello, ci si chiede se abbia senso usare le GAN nel contesto in cui operiamo, così come proposto da [Montavon et al., 2017]. Infatti, usare un modello complesso per spiegarne un altro è un po' come un cane che tenta di mordersi la coda. Nonostante ciò, comprendere come va modificato il framework per l'utilizzo con le GAN ci aiuterà a capire meglio come modificare AM per la tecnica che andremo a sviluppare.



Figura 2.7: Esempio di AM con l'utilizzo delle GAN, immagine tratta da [Nguyen et al., 2016a]

Il problema va ridefinito nel seguente modo:

$$\max_z \log p(\omega_c | g(z)) - \lambda ||z||^2 \quad (2.3)$$

Con z punto nello spazio latente Z , $g(z)$ modello generativo che mappa i punti in Z sullo spazio delle immagini del dataset. La differenza principale nel framework sta nel fatto che si ottimizza rispetto ad un punto nello spazio latente e non nel dominio delle immagini. Una volta individuata la soluzione ottima z^* al problema, il prototipo viene prodotto dando in input al modello generativo tale soluzione dello spazio latente:

$$x^* = g(z^*)$$

L'uso della Media nell'Activation Maximization

Tornando al problema 3.1 notiamo come, nonostante la regolarizzazione imposta, la tecnica produca immagini costituite per lo più da rumore. In realtà [Montavon et al., 2017] riescono ad ottenere dei prototipi molto più significativi

per MNIST. Infatti, ad una lettura più attenta del paper in questione si comprende che la reale quantità di interesse utilizzata dagli autori per la generazione dei prototipi è la seguente:

$$\max_x \log p(\omega_c|x) - \lambda||x - \hat{x}||^2 \quad (2.4)$$

Con \hat{x} immagine risultante dalla media delle immagini appartenenti alla classe ω_c . Il risultato dell'ottimizzazione con la suddetta quantità di interesse su MNIST produce i risultati in figura 2.8, sicuramente più validi di quelli ottenuti in precedenza.



Figura 2.8: Prototipi per MNIST, generati mediante AM, su Lenet-5 [Lecun et al., 1998], con inserimento della media nella regolarizzazione

L'utilizzo della media nella regolarizzazione consente di catturare meglio l'effettiva distribuzione dei dati e fa sì che tutti i prototipi siano praticamente uguali alla media stessa. Questo meccanismo funziona con MNIST poiché ne sfrutta le caratteristiche che sono ben rappresentate dalla media. Infatti, nel dataset in questione tutti i numeri sono centrati sulla base del proprio centro di massa e sono circondati solo da pixel neri. Inoltre, non ci sono problemi di prospettiva, rotazione o altre alterazioni delle immagini. La stessa situazione non si verificherebbe in un dataset complesso come Imagenet, per cui la varianza delle immagini all'interno della stessa categoria è molto più alta. Il tal senso l'uso della media è, in generale, sconsigliato.

Generazione di più prototipi.

Com'è possibile comprendere dalla discussione appena fatta, spesso non è possibile catturare in un'unica immagine la complessità di un modello, per questo si tende nella letteratura recente a sviluppare metodi che generano un set di prototipi. [Nguyen et al., 2016b] in particolare propongono la tecnica in *Figura 2.11*

per la generazione di un set di prototipi locali, in alternativa al singolo prototipo globale. Questa tecnica consente di catturare meglio la varianza di punti di vista dei soggetti in un dataset complesso e fornisce quindi rappresentazioni più accurate ed affidabili.

Algorithm 1 Multifaceted Feature Visualization

Input: a set of images U and a number of facets k

1. for each image in U , **compute** high-level (here fc7) hidden code Φ_i
2. **Reduce** the dimensionality of each code Φ_i from 4096 to 50 via PCA.
3. **Run** t-SNE visualization on the entire set of codes Φ_i to produce a 2-D embedding (examples in Fig. 4).
4. **Locate** k clusters in the embedding via k -means. for each cluster
5. **Compute** a mean image x_0 by averaging the 15 images nearest to the cluster centroid.
6. **Run** activation maximization (see Section 2.2), but **initialize** it with x_0 instead of a random image.

Output: a set of facet visualizations $\{x_1, x_2, \dots, x_k\}$.



Figura 2.9: Tecnica e risultato della stessa come mostrati in [Nguyen et al., 2016b]

Come accennato in precedenza, l’Activation Maximization può essere utilizzato anche nel contesto dell’interpretabilità locale. In tal senso, è necessario introdurre all’interno della quantità di interesse un fattore che consenta all’immagine generata di avvicinarsi ad uno specifico esempio. Uno dei modi per raggiungere questo obiettivo è il seguente:

$$\max_x \log p(\omega_c|x) - \eta \|x - x_0\|^2 \quad (2.5)$$

La quantità di interesse è molto simile alla 3.1, tranne per la presenza di x_0 nel termine di regolarizzazione. Quest’ultimo rappresenta un reference point per il quale si vuole fornire una spiegazione visiva. Il parametro η controlla la quantità di localizzazione.

In definitiva l’Activation Maximization è un framework di ottimizzazione estremamente flessibile. Consente di operare sia in modalità white-box che black-box, sia al fine di migliorare l’interpretabilità globale che quella locale. Sta all’utilizzatore della tecnica decidere in che modo sfruttarne le caratteristiche. Noi prediligiamo

remo, come già accennato, la modalità model-agnostic sia per l'interpretabilità globale che locale.

2.2.2 Sensitivity Analysis

Fa parte dei **salency methods**, una sottocategoria di visualization methods, che ha lo scopo di mostrare quali feature di un determinato input lo rendono rappresentativo di un certo concetto, che è risultato come output del modello. Nel caso delle immagini tali feature possono essere i pixel, mentre nel caso dei documenti di testo ogni parola può essere considerata una feature. Ad ogni elemento costitutivo dell'input verrà assegnato un punteggio di rilevanza che ne certificherà l'importanza nei confronti dell'output ottenuto dal modello.

Nello specifico la **sensitivity analysis** cerca di identificare le *feature* più importanti basandosi sui gradienti locali del modello, o su altre misure di variazione. Una possibile formulazione per la rilevanza è la seguente:

$$R_i(x) = \left(\frac{\partial f}{\partial x_i} \right)^2$$

dove $R_i(x)$ è il punteggio di rilevanza per la singola caratteristica x_i dell'input x . I gradienti possono essere facilmente calcolati nelle reti neurali utilizzando la **backpropagation**. Dunque, la sensitivity analysis opera nel contesto dell'interpretabilità locale e produce delle heatmap che mettono in risalto le zone dell'immagine più rilevanti al fine della classificazione.

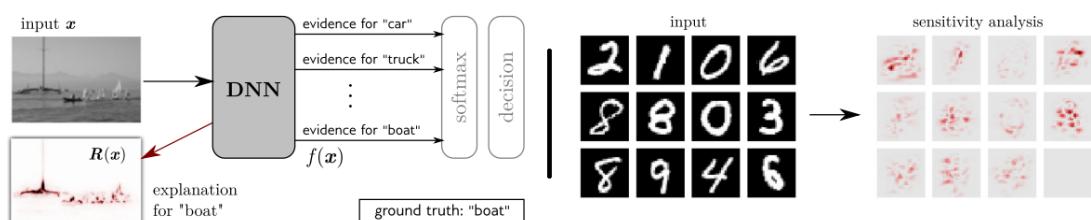


Figura 2.10: Immagine tratta da [Montavon et al., 2017]. A sinistra l'applicazione della sensitivity analysis ad una DNN per la spiegazione di un'immagine rappresentante una barca. A destra la stessa tecnica applicata ad una CNN sui digit di MNIST.

Nel corso degli anni sono state proposte molte varianti alla sensitivity analysis, che consentono di ottenere spiegazioni visive più accurate ed affidabili, tuttavia la maggior parte di esse ha necessità di sfruttare la struttura del modello per raggiungere l'obiettivo desiderato. Siccome la nostra intenzione è quella di rimanere model-agnostic non analizzeremo nello specifico queste varianti, vale tuttavia la pena citarle. Le principali alternative alla sensitivity analysis sono due: Deep Taylor Decomposition [Montavon et al., 2015] e PatternAttribution [Kindermans et al., 2017b].

L'inaffidabilità dei salency methods

I salency methods sono spesso stati oggetto di critica in letteratura. In particolare [Kindermans et al., 2017a] ritengono la maggior parte dei metodi in questa categoria poco affidabile e trovano paradossale il fatto che essi siano usati per proporre spiegazioni a modelli complessi, quando falliscono (in determinate condizioni) anche su semplici modelli di regressione lineare. [Kindermans et al., 2017a] muovono questa critica portando come prova il fatto che tali tecniche sono sensibili a fattori che non contribuiscono alla predizione del modello. Per dimostrare questa tesi gli autori utilizzano dei semplici step di preprocessing sulle immagini (e.g. mean shift dei dati in input) per mostrare che una trasformazione, senza effetto sul modello, può invece far fallire i salency methods. Per risolvere questo problema, [Kindermans et al., 2017a] propongono di valutare ed eventualmente modificare tali tecniche sulla base della *input invariance*. Quest'ultima è proposta come un'assioma e prevede che la sensibilità del salency method utilizzato debba riflettere la sensibilità del modello rispetto alla variazione dell'input.

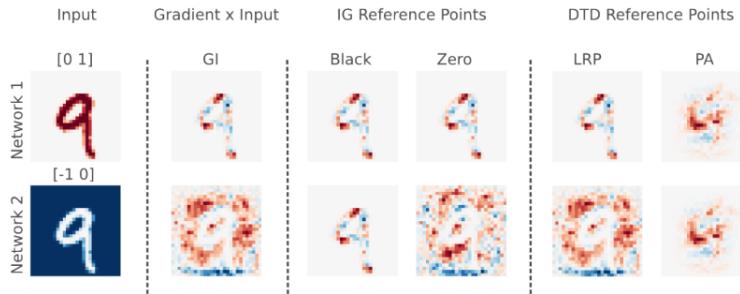


Figura 2.11: Immagine tratta da [Kindermans et al., 2017a]. Valutazione di vari attribution method per la sensitivity analisys su MNIST, su due modelli con codifiche diverse. Il primo (f1) nell’intervallo $[0,1]$, il secondo (f2) nell’intervallo $[-1,0]$. I metodi Gradient x Input, IG e LRP non sono affidabili e producono risultati diversi a seconda della rete. IG con un black reference point e PatternAttribution sono invarianti rispetto alla trasformazione dell’input.

Sempre gli stessi autori propongono due tecniche che sono corrette a livello teorico per i modelli lineari e scalabili su modelli complessi: PatterNet e PatternAttribution [Kindermans et al., 2017a] [Kindermans et al., 2017b].

2.2.3 Local Interpretable Model-agnostic Explanations - LIME

LIME è una tecnica di spiegazione model-agnostic proposta da [Ribeiro et al., 2016] come soluzione ai problemi d’interpretabilità locale e globale nel Machine Learning. Ad oggi è una delle soluzioni più utilizzate sia in campo accademico che industriale. Gli autori della tecnica propongono due soluzioni:

- LIME, un algoritmo che identifica un modello interpretabile (e.g. un modello di regressione lineare) localmente attorno ad una predizione che si intende spiegare.
- SP-LIME, un metodo per la selezione di istanze rappresentative, con le relative spiegazioni, per aumentare la fiducia nel modello. Gli autori propongono questa come soluzione al problema dell’interpretabilità globale.

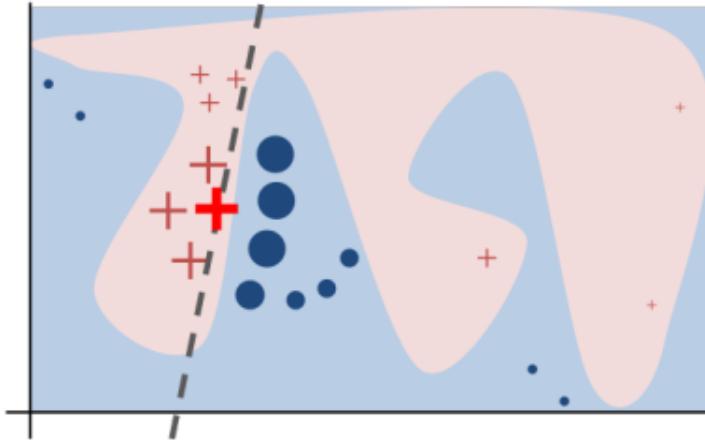


Figura 2.12: Immagine tratta da [Ribeiro et al., 2016]. Esempio dummy di un modello lineare appreso su un dataset di sample perturbati attorno all'istanza da spiegare. Localmente un modello lineare riesce ad approssimarne uno complesso.

La spiegazione prodotta da LIME è sempre di tipo visuale. Infatti, il sistema presenta all’utente artefatti testuali o visivi per fornire una comprensione qualitativa della relazione che intercorre tra le feature dei dati in input e la predizione del modello. Queste spiegazioni, sono comprese dagli utenti grazie alla loro conoscenza pregressa del dominio. Nel campo delle immagini tali spiegazioni assumono la forma di superpixel estrapolati dall’immagine che si intende spiegare. Nel campo dell’analisi testuale una rappresentazione interpretabile potrebbe essere un vettore binario che indica la presenza/assenza di una parola.

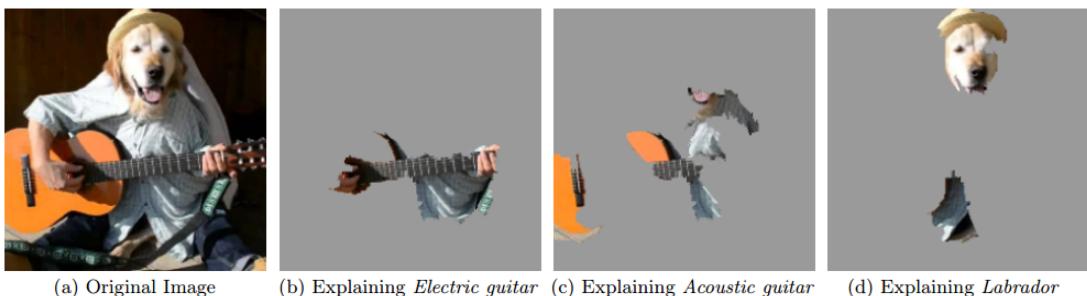


Figura 2.13: Immagine tratta da [Ribeiro et al., 2016]. Esempio di spiegazione per le Classi: Labrador, chitarra classica e chitarra elettrica

Siano $x \in R^d$ l’input del classificatore e $x' \in \{0, 1\}^{d'}$ il vettore binario usato per la sua rappresentazione interpretabile.

Come affermato in [Ribeiro et al., 2016] la spiegazione prodotta da LIME è otte-

nuta dall'ottimizzazione della seguente quantità:

$$\xi(x) = \arg \min_{g \in G} L(f, g, \pi_x) + \Omega(g) \quad (2.6)$$

Dove:

- $g \in G$ è il modello selezionato per la spiegazione a partire dall'insieme di modelli disponibili G (e.g. modelli lineari, decision trees..). Il dominio di g è $\{0, 1\}^{d'}$ (i.e. g agisce sull'assenza/presenza delle componenti interpretabili).
- $\Omega(g)$ è una misura della complessità del modello che si usa per la spiegazione (e.g. la profondità di un albero).
- $f : R^d \rightarrow R$ è il modello che deve essere spiegato e $f(x)$ la probabilità che l'istanza x appartenga ad una determinata classe (il sistema prende in considerazione una classe alla volta).
- Il termine $\pi_x(z)$ denota una misura di prossimità tra un'istanza z e x in modo da definire la località intorno ad x .
- $L(f, g, \pi_x)$ misura l'infedeltà della spiegazione g nell'approssimare f nella località π_x .

La tecnica punta a minimizzare la “locality aware loss” $L(f, g, \pi_x)$, mantenendo $\Omega(g)$ il più basso possibile, senza fare assunzioni sul modello f , visto che bisogna essere agnostici rispetto al modello.

Gli autori negli esperimenti portati avanti utilizzano la misura di prossimità in 2.7 e la loss in 2.8:

$$\pi_x = \exp(-D(x, z)^2 / \sigma^2) \quad (2.7)$$

$$L(f, g, \pi_x) = \sum_{z, z' \in Z} \pi_x(z)(f(z) - g(z'))^2 \quad (2.8)$$

Con D misura di distanza dipendente da dominio (e.g. nel dominio delle immagini si utilizza la distanza euclidea). Per risolvere il problema in questione, [Ribeiro et al., 2016] utilizzano il seguente algoritmo:

Algorithm 1 Sparse Linear Explanations using LIME

Require: Classifier f , Number of samples N
Require: Instance x , and its interpretable version x'
Require: Similarity kernel π_x , Length of explanation K

```

 $\mathcal{Z} \leftarrow \{\}$ 
for  $i \in \{1, 2, 3, \dots, N\}$  do
     $z'_i \leftarrow sample\_around(x')$ 
     $\mathcal{Z} \leftarrow \mathcal{Z} \cup \langle z'_i, f(z_i), \pi_x(z_i) \rangle$ 
end for
 $w \leftarrow K\text{-Lasso}(\mathcal{Z}, K)$   $\triangleright$  with  $z'_i$  as features,  $f(z)$  as target
return  $w$ 

```

Figura 2.14: Immagine tratta da [Ribeiro et al., 2016]. Algoritmo principale per la generazione della spiegazione mediante LIME

La tecnica fino a questo punto consente di rispondere al problema dell’interpretabilità locale. Tuttavia, per valutare il modello nel complesso non è sufficiente fornire la spiegazione per una singola predizione. Gli autori propongono quindi di aumentare la comprensione del modello tramite un set di spiegazioni. Questo approccio risulterà complementare alle metriche usuali, come l’accuratezza. Si pone quindi ora il problema di come scegliere le istanze, soprattutto in grandi dataset.

Si rappresenti il tempo/pazienza che gli umani hanno con un budget B , che indica il numero di spiegazioni che sono disposti a esaminare per comprendere il modello. Dato un set di istanze X , si definisce un “pick step” come il task di selezionare B istanze che l’utente dovrà analizzare. Date le spiegazioni per un set di istanze X si costruisce una “explanation matrix” W di dimensioni $n \times d'$ che rappresenta l’importanza locale delle componenti interpretabili per ogni istanza. Inoltre, per ogni colonna j di W , I_j denota l’importanza globale che ha quella componente nello spazio delle spiegazioni. L’algoritmo per SP-LIME e le equazioni necessarie all’elaborazione sono presentate di seguito:

Algorithm 2 Submodular pick (SP) algorithm

Require: Instances X , Budget B

```
for all  $x_i \in X$  do
     $\mathcal{W}_i \leftarrow \text{explain}(x_i, x'_i)$            ▷ Using Algorithm 1
end for
for  $j \in \{1 \dots d'\}$  do
     $I_j \leftarrow \sqrt{\sum_{i=1}^n |\mathcal{W}_{ij}|}$    ▷ Compute feature importances
end for
 $V \leftarrow \{\}$ 
while  $|V| < B$  do
     $V \leftarrow V \cup \text{argmax}_i c(V \cup \{i\}, \mathcal{W}, I)$ 
end while
return  $V$ 
```

$$c(V, \mathcal{W}, I) = \sum_{j=1}^{d'} \mathbb{1}_{[\exists i \in V : \mathcal{W}_{ij} > 0]} I_j$$

$$\text{Pick}(\mathcal{W}, I) = \underset{V, |V| \leq B}{\text{argmax}} c(V, \mathcal{W}, I)$$

Figura 2.15: Immagine tratta da [Ribeiro et al., 2016]. SP-LIME: algoritmo per migliorare l'interpretabilità globale di un modello.

Capitolo 3

Un nuovo approccio al problema dell'interpretabilità

Il seguente capitolo è il fulcro della dissertazione. Innanzitutto, qui esporremo i principi di base sui quali si fonda la tecnica proposta, entrando nel dettaglio del perché si è ritenuto opportuno sviluppare questo approccio alternativo al problema dell'interpretabilità. Successivamente, forniremo un overview dell'architettura del sistema, con particolare enfasi sulla forte modularità dello stesso. Analizzeremo, quindi, le tecniche di *dictionary learning* che rappresentano il principale tratto distintivo del nostro approccio. Infine, entreremo nei dettagli implementativi della tecnica e tratteremo le tecnologie necessarie alla sua realizzazione.

3.1 L'idea di base e l'architettura del sistema

Nella sezione 1.6 abbiamo individuato dei principi, sulla cui base, impostare una nuova metodologia per affrontare il problema della spiegazione delle risposte dei modelli di machine learning, con particolare enfasi sui sistemi di classificazione delle immagini. In breve, intendiamo creare un metodo che risulti agnostico rispetto al modello che tenta di spiegare, che si basi su un ragionamento logico rigoroso e che sia in grado di fornire risultati sia per l'interpretabilità locale che globale del modello in esame. Viene quindi da chiedersi se tale metodo non esista già in letteratura. In sezione 2 abbiamo fatto una disamina dei metodi più

utilizzati per aumentare l’interpretabilità di un modello. In particolare, abbiamo scelto di focalizzarci su tre metodi: Activation Maximization, sensitivity analysis e LIME. Questo tipo di analisi ha avuto uno scopo ben preciso: porre le fondamenta per lo sviluppo di un approccio innovativo al problema dell’interpretabilità. Da ogni metodo preso in considerazione, abbiamo estratto le caratteristiche ritenute più interessanti e vantaggiose. Nello specifico, la tecnica che andiamo a proporre deve, all’Activation Maximization la struttura generale del framework di ottimizzazione, ai salency methods il concetto di rilevanza associato alle feature dell’input, a LIME il concetto di superpixel da mostrare all’utente. Se, come abbiamo visto, esistono già delle tecniche per migliorare l’interpretabilità dei modelli per la classificazione delle immagini, da cosa nasce l’esigenza di un nuovo approccio al problema? Durante l’analisi del framework AM ci si è accorti di una differenza fondamentale di trattamento delle feature tra la classificazione di documenti testuali e quella delle immagini. Nei primi le caratteristiche sulla cui base generare una spiegazione sono parole di senso compiuto e pertanto risultano di facile comprensione per un utente. Nel secondo caso, invece, si utilizzano i pixel di un’immagine, ma un pixel non porta con sé nessun contenuto informativo per l’utente.

Per essere più chiari, forniamo un esempio. Se giustifichiamo la classificazione di un documento relativo al cibo sulla base della presenza/assenza di parole come ‘pasta’, ‘pane’, ‘acqua’, ‘torta’, ecc., l’utente sarà in grado di comprendere tale spiegazione, poiché composta da strutture che hanno un significato intrinseco. Se spieghiamo la classificazione dell’immagine di un gatto mediante la presenza/assenza di uno o più pixel in determinate posizioni dell’immagine, non forniamo nessun indizio concreto all’utente. Questo perché, come affermato, affinché si possa attribuire un significato qualitativo ai pixel è necessario che essi siano agglomerati per creare forme, texture ed in generale strutture comprensibili.

Nel mezzo del
cammin di nostra vita
mi ritrovai per una
selva oscura,
ché la diritta via era
smarrita.

vita



Figura 3.1: Esempio di feature in un documento, a sinistra. A destra esempio di feature in un’immagine.

In tal senso riteniamo che l’uso di AM in letteratura si concentri eccessivamente sulla bontà con cui viene ricostruita/generata un’immagine. A nostro avviso, l’obiettivo da raggiungere dovrebbe essere quello di individuare patch di un’immagine che siano particolarmente rilevanti alla sua classificazione su un determinato concetto. In realtà, come mostreremo in sezione 3.1.2, è possibile utilizzare questo framework versatile come base per la selezione di tali strutture. Sebbene l’obiettivo finale sia simile a quello di LIME, che mostra agli utenti dei superpixel appartenenti all’immagine da spiegare, il nostro approccio è differente. Infatti, invece di perturbare l’immagine di partenza, puntiamo a costruire un dizionario di elementi grafici significativi, a partire dal dataset che si è utilizzato per l’apprendimento del modello. Così come nei testi si ha a disposizione (seppur in modo隐式) un vocabolario di parole, noi vogliamo un vocabolario di strutture di senso compiuto per gli esseri umani. Qui entra in gioco il dictionary learning, che discutiamo in sezione 3.1.1, il quale consente di apprendere il dizionario in questione. La valutazione della bontà degli elementi generati sarà prettamente qualitativa, ma cercheremo di ottenere i risultati desiderati imponendo dei vincoli, per lo più di sparsità, agli elementi prodotti.

Dunque, ricapitolando, l’obiettivo della nostra tecnica non sarà la ricostruzione di un’immagine, ma l’individuazione di patch fondamentali alla descrizione del concetto che si sta classificando, così come mostrato in *Figura 3.2*.

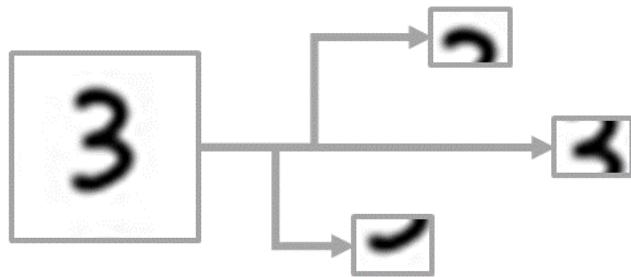


Figura 3.2: Idea di base del sistema. Spiegazione del perché il modello in questione ha classificato l’immagine come un 3.

Dopo aver esposto i principi alla base della tecnica che intendiamo sviluppare, impostiamo ora un ragionamento che ci consenta di definire, almeno a grandi linee, l’architettura del sistema. Ci concentreremo dapprima sul problema dell’interpretabilità locale e successivamente prenderemo in considerazione anche la globale. Innanzitutto, per realizzare l’idea alla base della tecnica, necessitiamo di alcuni elementi fondamentali, che andiamo ad elencare:

- Un’immagine da dare in input al modello per ricevere una classificazione della stessa. Oppure, una classe per la quale si vogliono generare i prototipi.
- Un modello di classificazione per le immagini.
- Un dizionario di elementi strutturati (i.e. patch di immagini), ottenuto tramite dicitonary learning.
- Un metodo che ci consenta di selezionare gli elementi dal dizionario più rilevanti ai fini della spiegazione visiva dell’immagine (oppure al fine di generare i prototipi per la classe scelta).

Questi fattori si fondono in modo coerente tra loro, generando una struttura modulare e vantaggiosa. La figura 3.3 riassume in modo visivo l’architettura proposta.

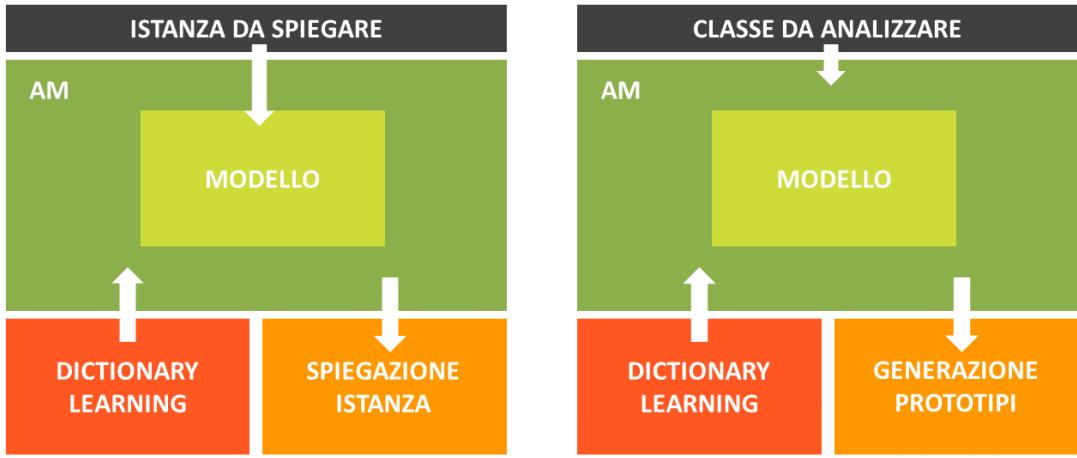


Figura 3.3: Architettura del sistema. A sinistra un framework per l’interpretabilità locale. A destra lo stesso framework per l’interpretabilità globale.

Siccome le due tecniche sono molto simili, ci concentreremo per il resto del capitolo sul problema dell’interpretabilità locale.

Possiamo quindi scindere la tecnica in due fasi principali, che andiamo ad analizzare.

3.1.1 Fase 1 - Apprendimento del dizionario mediante Dictionary Learning

Come affermato nella sezione precedente la prima fase dell'algoritmo si occupa della generazione di un dizionario di strutture di base. Questo obiettivo è raggiunto impiegando determinati algoritmi di dictionary learning. Ma in cosa consiste, in generale, questa tecnica? Il dictionary learning è un metodo di fattorizzazione matriciale utilizzato per la modellazione di segnali. Un segnale può assumere varie forme, noi ci concentreremo sul dominio delle immagini.

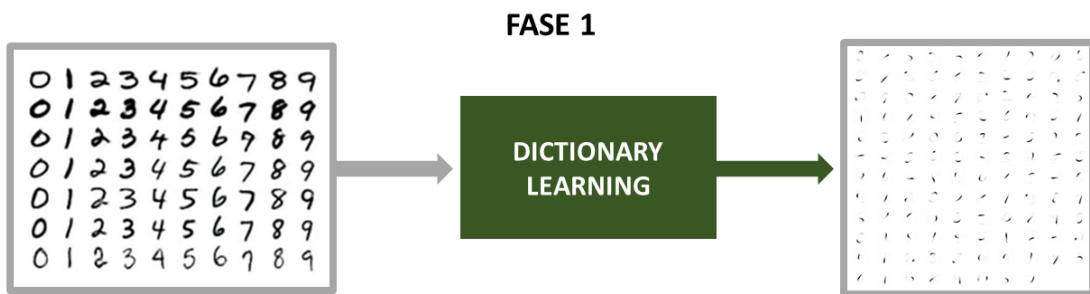


Figura 3.4: Fase 1 della tecnica sviluppata, generazione di un dizionario di strutture di base, dette atomi.

L'obiettivo del dictionary learning è, quindi, quello di approssimare un segnale mediante la combinazione lineare di altri segnali di base. Formalmente:

$$v \approx W \times h$$

Con v segnale da approssimare (nel nostro caso immagine), W dizionario e h vettore di coefficienti (detto anche encoding).

Uno degli elementi fondamentali nell'ambito del dictionary learning è, ovviamente, il dizionario. Quest'ultimo è una matrice di elementi di base, detti anche segnali prototipi o atomi. Se indichiamo la dimensionalità del segnale v da approssimare con N (e.g. un'immagine 8x8 ha una dimensionalità di 64), allora il dizionario sarà una matrice di $N \times K$ elementi tali per cui:

- se $K > N$ il dizionario è detto *overcomplete*
- se $K = N$ il dizionario è detto *complete*

- se $K < N$ il dizionario è detto *undercomplete*

Nelle affermazioni fatte fino a questo momento il dizionario è stato considerato come un oggetto immutabile, dato per assodato, di cui si può disporre per la ricostruzione dei segnali. In realtà esso va appreso a partire da un dataset di partenza. In figura 3.5 è mostrato tale concetto in modo schematico. V rappresenta il dataset, W il dizionario e H l'encoding (matrice e non più vettore per ricostruire tutti gli elementi del dataset di partenza).

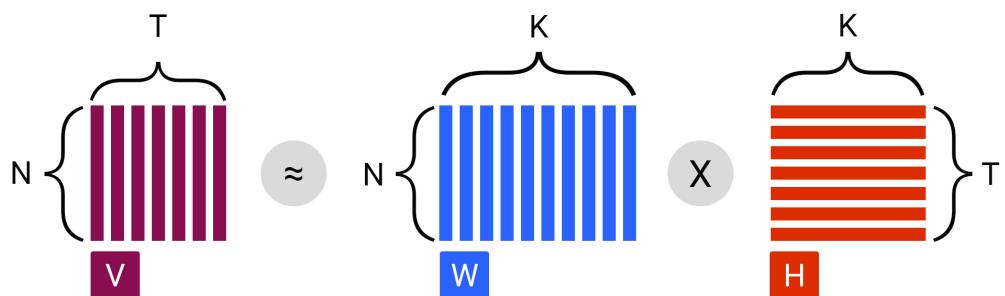


Figura 3.5: Rappresentazione grafica di una tecnica di dictionary learning. V rappresenta il dataset di partenza, costituito da T elementi di dimensionalità N . W è il dizionario che si va ad apprendere, formato da K atomi di dimensionalità N . H è l'encoding, ossia la matrice dei pesi necessaria a ricostruire il dataset di partenza, mediante un prodotto vettoriale con il dizionario. H è una matrice di T elementi di dimensionalità K .

Il procedimento di apprendimento del dizionario si basa sulla valutazione di una misura di distanza tra la matrice che rappresenta il dataset di riferimento e quella data dal prodotto del dizionario per l'encoding. In letteratura sono state proposte varie misure, una tra le più utilizzate è sicuramente il quadrato della norma di Frobenius, sulla cui base possiamo ridefinire il problema del dictionary learning nel seguente modo:

$$\min_{W,H} \|V - WH\|_F$$

Minimizzare tale quantità non è un compito banale, poiché la funzione risulta non convessa se si analizzano W e H contemporaneamente.

[Olshausen and Field, 1997] sono stati i primi a proporre uno schema risolutivo

per tale problematica. Gli autori sono partiti dalla seguente considerazione: dal momento in cui la suddetta funzione risulta convessa solo in W o solo in H , la minimizzazione può essere effettuata alternando le ottimizzazioni rispetto al dizionario ed ai coefficienti. Questa soluzione è diventata lo standard de facto per la risoluzione di problemi di dictionary learning. Da essa derivano importanti algoritmi come K-SVD che risolve il problema del dictionary learning cui sono stati aggiunti dei vincoli di sparsità sull'encoding.

Dunque, tramite tecniche di dictionary learning, siamo in grado di apprendere un dizionario di strutture visive sulla cui base è possibile generare una spiegazione comprensibile per un utente. Vale la pena, inoltre, ricordare che le tecniche di dictionary learning sono una estensione della Principal Component Analysis (PCA) quando sono rimossi alla stessa i due seguenti vincoli [Tessitore and Prevete, 2011]:

- Atomi ortonormali
- Numero atomi uguale alla dimensionalità del segnale

Nella sezione 3.2 metteremo in luce dei criteri per la scelta della tecnica di dictionary learning, e indicheremo il funzionamento di una soluzione in particolare. Una volta ottenuto il dizionario, è opportuno capire quali elementi dello stesso siano più rilevanti alla classificazione. È necessario dunque un algoritmo per la selezione di tali strutture. Tale metodo è esposto nella seconda fase della tecnica, *Sezione 3.1.2*.

3.1.2 Fase 2 - Selezione degli elementi strutturali necessari alla spiegazione dell'istanza

La fase 1 dell'algoritmo produce un dizionario. A partire da quest'ultimo sarà necessario impostare una procedura che ci consenta di selezionare gli atomi più rilevanti ai fini della spiegazione che vogliamo proporre. Per impostare un ragionamento strutturato, sfruttiamo il framework messo a disposizione dall'Activation Maximization e lo modifichiamo in modo da riuscire a raggiungere il nostro

obiettivo. Ricordiamo che nella sua versione più basilare AM opera al fine di ottimizzare la seguente quantità di interesse:

$$\max_x \log p(\omega_c|x) - \lambda \|x\|^2 \quad (3.1)$$

L'ottimizzazione avviene mediante ascesa del gradiente su un modello già addestrato, per il quale sono bloccati tutti i parametri. Per questo motivo le feature dell'input, ossia i pixel dell'immagine, sono le uniche variabili da cui dipende il valore della quantità di interesse. In tal senso, ricordiamo che x è l'immagine generata come risultato dell'ottimizzazione e che va in pasto al modello per produrre la quantità $\log p(\omega_c|x)$, mentre $\lambda \|x\|^2$ è un termine di regolarizzazione che impone una preferenza per gli input vicini all'origine. Per attuare la tecnica che proponiamo è necessario modificare la quantità di interesse nel seguente modo:

$$\max_h \log p(\omega_c|Wh) - \lambda_1 \|Wh - i\|_2 + \lambda_2 sparsity(h) \quad (3.2)$$

Con $W \times h = x$ che rappresenta l'immagine ricostruita come combinazione lineare degli atomi presenti nel dizionario W e del vettore dei pesi (encoding) h , i immagine da spiegare, $sparsity(h)$ misura di sparsità del vettore dei pesi. Quindi ricapitolando, dato un modello già addestrato ed un dizionario calcolato nella prima fase dell'algoritmo:

1. Si genera un vettore dei pesi (o encoding) in modo random.
2. Si effettua un prodotto vettoriale con il dizionario in modo da generare un'immagine $x = W \times h$.
3. Si calcola la quantità di interesse. Ossia, si fornisce in input al modello l'immagine $x = W \times h$ e si ottiene una probabilità di classificazione e si calcola la distanza euclidea tra l'immagine ricostruita e quella originale.
4. Si propaga l'errore, mediante ascesa del gradiente, sull'encoding h .
5. Si torna al passo (2).

In Figura 3.7 è riassunto il ragionamento appena effettuato.

FASE 2

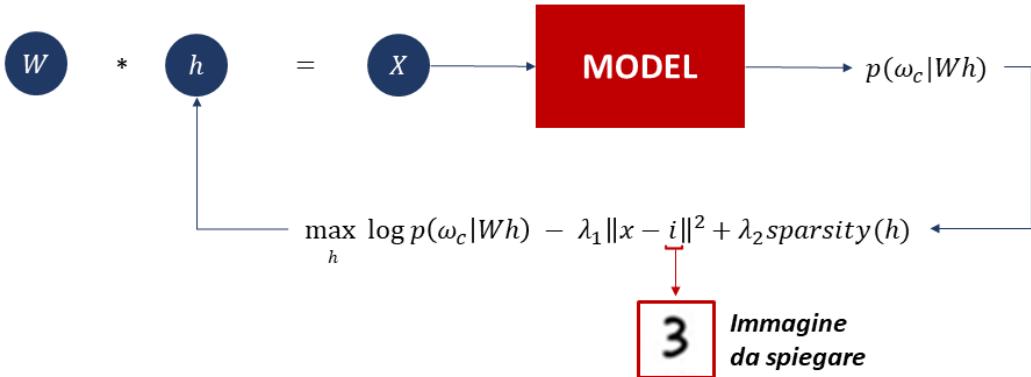


Figura 3.6: Fase 2 della tecnica sviluppata. Immagine riassuntiva. Interpretabilità locale.

Dopo aver compreso come modificare AM vale la pena soffermarsi su alcuni aspetti della quantità di interesse proposta in 3.2. La differenza principale con AM, sta nel fatto che ora ottimizziamo rispetto ad h (il vettore di pesi associato al dizionario) e non più rispetto ad x . Si passa quindi dalla generazione di un’immagine come risultato di un processo di ottimizzazione alla selezione degli atomi del dizionario più rilevanti al nostro scopo. Il primo termine di regolarizzazione, $\lambda_1 \|x - i\|_2$ è una penalità, basata su una misura di distanza tra immagine reale e ricostruita, che porta il processo a scegliere atomi, mediante la codifica h , in grado di generare una rappresentazione che si avvicina all’immagine originaria. Sebbene, come affermato in precedenza, la ricostruzione dell’immagine originale sia non necessaria, in questo contesto risulta utile per indirizzare il processo di ottimizzazione verso la ricerca degli atomi corretti. Il secondo criterio di regolarizzazione si basa su una misura di sparsità, $\lambda_2 \text{sparsity}(h)$. Questo perché, come osservato nel Capitolo 1, uno dei fattori di comprensibilità della spiegazione è relativo alla complessità della stessa. Imponendo la sparsità sull’encoding siamo in grado di limitare il numero di atomi selezionati dal processo di ottimizzazione e questo consente di avere idealmente, in output, spiegazioni più concise e chiare. Sul criterio di sparsità da scegliere ci soffermeremo molto nelle prossime sezioni.

ni, ma possiamo già asserire che esso è fortemente dipendente dalla tecnica di Dictionary Learning selezionata.

3.2 Scelta della tecnica di Dictionary Learning

In *Sezione 1.5* abbiamo individuato dei principi fondamentali che una spiegazione dovrebbe avere per essere efficace. Tra questi un ruolo di primo piano è ricoperto dalla *parsimonia*, secondo la quale una spiegazione dovrebbe essere costituita solo da pochi elementi fondamentali. In tal senso, la tecnica di Dictionary Learning che si decide di utilizzare dovrebbe prendere in considerazione il concetto di sparsità, sia per quanto riguarda gli atomi del dizionario, sia per quanto concerne l'encoding. Sempre riferendoci al campo del processing delle immagini, nel primo caso diremo che un atomo di un dizionario risulta sparso se è costituito da pochi pixel con valore diverso da zero. Idealmente, la sparsità non dovrebbe intaccare la struttura. Infatti, non ha molto senso avere un atomo con pochi pixel, ma sparsi su tutta l'immagine. Quello che vogliamo è un atomo con pochi pixel che formano una struttura con un significato, almeno qualitativo, per l'utente.

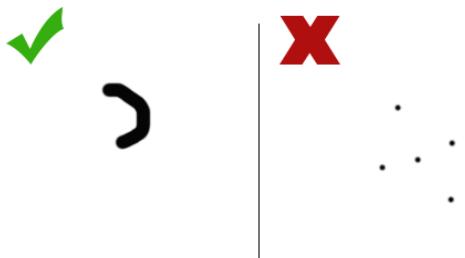


Figura 3.7: Esempio di struttura sparsa che intendiamo generare a sinistra. A destra un esempio di sparsità non corretta.

Nel secondo caso, diremo che il vettore dei pesi risulta sparso se pochi elementi dello stesso sono diversi da zero, questo significa infatti che stiamo effettivamente selezionando pochi atomi dal dizionario per generare la spiegazione. In tale contesto non ci interessa che gli atomi selezionati siano 'vicini' tra loro. Un altro criterio di scelta, che ha influenzato fortemente lo sviluppo della tecnica, è quello della *positività*. Nel nostro caso abbiamo scelto una tecnica di Dictionary

Learning che imponge un vincolo di positività sia sugli atomi del dizionario, sia sull’encoding. Questa decisione potrebbe sembrare, in un primo momento, senza alcun fondamento. In realtà, essa si basa su alcune evidenze psicologiche e teorie computazionali che ritengono la percezione del tutto basata sulla percezione delle sue parti [Andrew, 1997], [Biederman, 1987], [Logothetis and Sheinberg, 1996], [Wachsmuth et al., 1994], [Palmer, 1977]. Imporre un vincolo di positività ci consente di effettuare solo operazioni additive tra gli atomi, questo aumenta la comprensibilità della spiegazione da parte dell’utente. Infatti, operazioni complesse di cancellazione dei pixel, potrebbero aumentare la capacità di approssimazione della tecnica utilizzata, ma sono difficili da immaginare. A differenza di queste ultime, le operazioni additive sono facili da comprendere e aumentano la chiarezza della spiegazione, altro fattore di eccezionale importanza. Altro criterio di scelta, come notato in precedenza, è la *struttura* che gli atomi devono possedere. Se essi vanno a formare delle chiazze di colore senza senso, non possono essere utilizzati ai fini del metodo proposto. Vanno quindi generati degli atomi che siano, almeno dal punto di vista qualitativo, ben strutturati e comprensibili già come entità autonome. Non basta infatti che lo siano quando sono combinati in modo lineare. Gli algoritmi di dictionary learning che andremo ad utilizzare non impongono formalmente una struttura, quindi non è garantito che la decomposizione abbia un valore semantico valido [Laroche et al., 2015]. Tuttavia, come mostrato in [Hoyer, 2004], all’atto pratico essi riescono a generare dizionari con elementi ben strutturati. Sulla base delle considerazioni fatte, la nostra scelta è ricaduta su due algoritmi: Non-negative Matrix Factorization (NMF) e NMF con vincoli di sparsità. Di seguito presentiamo le due metodologie e poi proponiamo un’implementazione della tecnica in esame.

3.2.1 NMF (Non-negative Matrix Factorization)

È un algoritmo per la fattorizzazione di matrici ($V \approx WH$) che impone un vincolo di non negatività sia sul dizionario che sull’encoding. Formalmente definiamo il seguente problema:

Minimizzare $\|V - WH\|_F$ rispetto a W e H , con $W, H \geq 0$

Come affermato in [Lee and Sebastian Seung, 1999] il vincolo di non negatività è ciò che differenzia quest'algoritmo da altri simili, come PCA e VQ. Questi ultimi apprendono rappresentazioni olistiche, non basate sull'unione delle parti. NMF invece, consentendo solo combinazioni additive, implementa effettivamente una costruzione del tutto basata sull'insieme delle sue parti. Ciò rende le singole componenti interpretabili al di là della loro combinazione.

Inoltre, le basi e gli encoding di NMF contengono una grossa frazione di “vanishing coefficients”, per cui entrambi possono essere considerati sparsi.

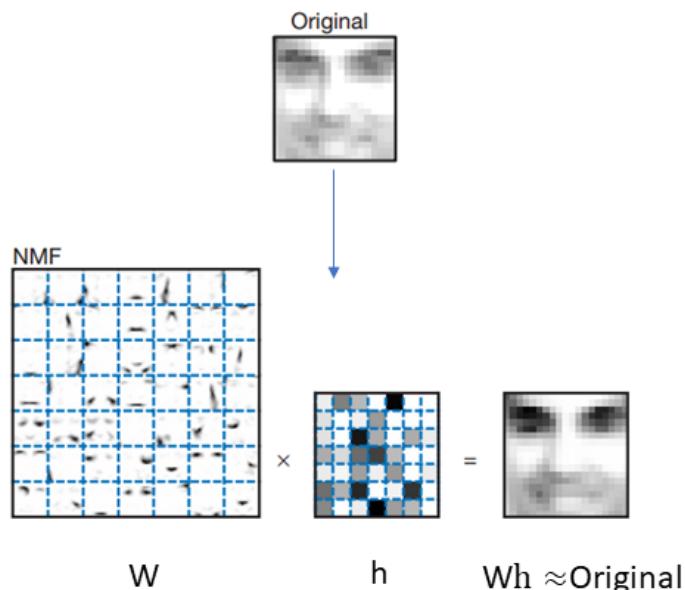


Figura 3.8: NMF - immagine tratta da [Lee and Sebastian Seung, 1999]. W rappresenta il dizionario, in questo caso gli atomi sono rappresentati in modo visuale. Per questo motivo anziché essere vettori sono immagini. Allo stesso modo h , ossia l'encoding necessario alla ricostruzione dell'immagine originaria è espresso come matrice.

Gli algoritmi risolutivi alla base di NMF prendono spunto dallo schema proposto da [Olshausen and Field, 1997]. In particolare, [Lee and Seung, 2000] hanno messo a punto delle regole di aggiornamento moltiplicative per W ed H , facili da implementare ed estremamente eleganti. Essi propongono il seguente teorema, che dimostrano nell'articolo appena citato.

Teorema. La norma di Frobenius $\|V - WH\|_F$ è non crescente sotto le seguenti regole

$$H_{\alpha\mu} \leftarrow H_{\alpha\mu} \frac{(W^T V)_{\alpha\mu}}{(W^T W H)_{\alpha\mu}} \quad W_{i\alpha} \leftarrow W_{i\alpha} \frac{(V H^T)_{i\alpha}}{(W H H^T)_{i\alpha}}$$

La norma di Frobenius risulta invariante se sottoposta a questi aggiornamenti se e solo se W e H sono in un punto stazionario della distanza.[Lee and Seung, 2000]

3.2.2 NMF con vincoli di sparsità

La sparsità in NMF è una conseguenza, ma non viene imposta con dei vincoli. Ci sono sviluppi di dataset per i quali l'uso di NMF non produce risultati soddisfacenti, in termini di rappresentazioni locali dei dati. Infatti, è stato mostrato che su alcuni dataset di immagini di volti, il dizionario generato da NMF è costituito da atomi contenenti rappresentazioni globali, anziché locali. Per ovviare a tale problematica, sono state sviluppate delle varianti di NMF che rafforzano la sparsità e, idealmente, riescono ad ottenere sempre atomi sparsi e quindi locali. Una di queste è “NMF with sparseness constraints” [Hoyer, 2004], che analizziamo di seguito.

[Hoyer, 2004] propone una misura di sparsità di un vettore basata sul rapporto tra la norma L_1 e la norma L_2 dello stesso.

$$\text{sparseness}(x) = \frac{\sqrt{n} - (\sum |x_i|)/\sqrt{\sum x_i^2}}{\sqrt{n} - 1}$$

Con x vettore di cui si valuta la sparsità e n dimensionalità di x . Tale misura esibisce le seguenti proprietà:

- Assume valore unitario se e solo se x contiene solo una singola componente diversa da 0.
- Assume valore nullo se e solo se tutte le componenti sono uguali.
- Interpola in modo smooth tra i due estremi precedenti.

Sulla base della suddetta misura è possibile definire “NMF with sparseness constraints”:

Definizione: NMF with sparseness constraints

Data una matrice non negativa V di dimensioni $N \times T$, trovare le matrici non negative W e H , rispettivamente di dimensioni $N \times K$ e $K \times T$ tali che:

$$E(W, H) = \|V - WH\|_F$$

risulti minima, sotto i seguenti vincoli:

$$\text{sparseness}(w_i) = S_w \quad \forall i$$

$$\text{sparseness}(h_i) = S_h \quad \forall i$$

dove w_i è la i -esima colonna di W e h_i è la i -esima riga di H .

K indica il numero di atomi del dizionario, mentre S_w e S_h i livelli di sparsità desiderata.

Un framework “generico” Il problema principale che si presenta quando si impone la sparsità su encoding o dizionario mediante norma L_1 o L_0 è che queste ultime non sono derivabili. Non si può quindi applicare una tecnica di ottimizzazione come la discesa del gradiente alla funzione nel suo complesso. In questo caso si fa spesso ricorso ai proximal methods [Combettes and Pesquet, 2011]. Il ragionamento generico è il seguente. Data una quantità da minimizzare, ad esempio

$$E(W, H) = \|V - WH\|_F + \|H\|_1$$

Essa può essere scissa in una parte derivabile, e quindi ottimizzabile mediante discesa del gradiente ed una parte non derivabile. Possiamo quindi riscrivere la quantità precedente come segue:

$$E = F(X) + J(X)$$

con

$$F(X) = \|V - WH\|_F$$

parte differenziabile, e

$$J(X) = \|H\|_1$$

parte non differenziabile.

La prima parte $F(X)$ può essere ottimizzata appunto mediante discesa del gradiente, sempre seguendo lo schema che alterna l’ottimizzazione di W e H . La derivata della seconda parte $J(X)$ può essere approssimata mediante un operatore di proiezione, nel caso della norma L_1 l’operatore in questione è la soft-threshold[Basso et al., 2010].

Seguendo il ragionamento appena delineato, [Hoyer, 2004] implementa l’algoritmo a pagina seguente:

Algoritmo 3.1: NMF with sparseness constraints

```
1: function NMFHoyer(V,Sw,Sh,N,K,T,maxIter,sL1,sL2)
2:    $W \leftarrow \text{random}(N, K)$ 
3:    $H \leftarrow \text{random}(K, T)$ 
4:   if ( $\text{Sw} \neq 0$ ) then
5:     for all  $column \in W$  do
6:        $W \leftarrow \text{proj}(column, sL1, sL2)$ 
7:     end for
8:   end if
9:   if ( $\text{Sh} \neq 0$ ) then
10:    for all  $row \in H$  do
11:       $H \leftarrow \text{proj}(row, sL1, sL2)$ 
12:    end for
13:  end if
14:   $iterCount \leftarrow 0$ 
15:  repeat
16:    if ( $\text{Sw} \neq 0$ ) then
17:       $W \leftarrow W - \mu_W(WH - V)H^T$ 
18:      for all  $column \in W$  do
19:         $W \leftarrow \text{proj}(column, sL1, sL2)$ 
20:      end for
21:    else
22:       $W \leftarrow W \otimes (VH^T) \oslash (WHH^T)$ 
23:    end if
24:    if ( $\text{Sh} \neq 0$ ) then
25:       $H \leftarrow H - \mu_HW^T(WH - V)$ 
26:      for all  $row \in H$  do
27:         $H \leftarrow \text{proj}(row, sL1, sL2)$ 
28:      end for
29:    else
30:       $H \leftarrow H \otimes (W^TV) \oslash (W^TWH)$ 
31:    end if
32:     $iterCount \leftarrow iterCount + 1$ 
33:  until convergence  $\vee iterCount > maxIter$ 
34:  return  $W, H$ 
35: end function
```

Le linee 1 e 2 inizializzano le matrici W e H di dimensioni rispettivamente $N \times K$ e $K \times T$. Successivamente, se ci sono vincoli di sparsità su W e/o H si impongono tali constraint mediante un operatore di proiezione (che analizzeremo nel dettaglio in seguito). In breve, l'operatore proietta ogni colonna di W (o riga di H) in modo che sia non negativa, con norma L_2 invariata e norma $L1$ settata in modo da ottenere la sparseness desiderata. Dopo una fase di inizializzazione, inizia un ciclo che ha come condizione di uscita la convergenza dell'algoritmo (i.e. $E(W, H) \leq \epsilon$) oppure un numero massimo di iterazioni prestabilite. All'interno di questo ciclo si ha il nucleo dell'algoritmo. La linea 15 controlla che vi sia la necessità di imporre sparsità su W , in tal caso si effettua un passo di una classica discesa del gradiente e si proiettano le colonne della matrice come in precedenza. Se non è richiesta la sparsità, si utilizzano le regole moltiplicative proposte da [Lee and Seung, 2000]. La stessa operazione viene ripetuta per H . I simboli \otimes e \oslash indicano rispettivamente moltiplicazione e divisione elementwise.

L'operatore di proiezione utilizzato nell'algoritmo appena descritto è stato messo a punto dallo stesso autore e consente di imporre il livello di sparsità (in relazione alla misura precedentemente definita) desiderato sulle matrici. Di seguito un'overview dell'operatore.

Algoritmo 3.2: Operatore di proiezione

```
1: function PROJ(X,L1,L2)
2:   for  $i \leftarrow 0, x.length$  do
3:      $s_i \leftarrow x_i + (L_1 - \sum x_i)/dim(x)$ 
4:   end for
5:    $Z \leftarrow \{\}$ 
6:   while true do
7:     if  $i \notin Z$  then
8:        $m_i \leftarrow L_1/(dim(x) - size(Z))$ 
9:     else
10:       $m_i \leftarrow 0$ 
11:    end if
12:     $s \leftarrow m + \alpha(s - m)$             $\triangleright \alpha \geq 0$  scelto in modo da soddisfare  $L_2$ 
13:    if  $s \geq 0$  then
14:      return  $s$ 
15:    end if
16:    for  $i \leftarrow 0, s.length$  do
17:      if  $i < 0$  then
18:         $Z \leftarrow Z \cup \{i\}$ 
19:      end if
20:    end for
21:    for  $i \in Z$  do
22:       $s_i \leftarrow 0$ 
23:    end for
24:     $c \leftarrow (\sum s_i - L_1)/(dim(x) - size(Z))$ 
25:    for  $i \notin Z$  do
26:       $s_i \leftarrow s_i - c$ 
27:    end for
28:  end while
29: end function
```

L'algoritmo esposto opera nel seguente modo. Innanzitutto, proietta il vettore in input sull'iperpiano $\sum s_i = L_1$. Successivamente, all'interno di questo spazio, effettua una proiezione verso il punto più vicino sull'ipersfera data dall'intersezione dei vincoli di somma ed L_2 . Se il risultato di quest'operazione è un vettore completamente non negativo, si è ottenuto un risultato valido, altrimenti si fissano le componenti negative del vettore a 0 e si ripetono le operazioni precedenti [Hoyer, 2004]. È stato dimostrato dall'autore del paper in questione che per l'operatore di proiezione proposto, sebbene il numero massimo di iterazioni sia pari alla dimensione del vettore x , nella pratica l'algoritmo converge in poche iterazioni, soprattutto su vettori con molte dimensioni.

3.2.3 Implementazione della nuova tecnica

All’inizio di questo capitolo è stato proposto un nuovo approccio al problema dell’interpretabilità per sistemi di classificazione di immagini. Abbiamo poi analizzato l’architettura generale del sistema, focalizzandoci particolarmente sulla modularità della stessa. In seguito, abbiamo definito in modo più preciso i passi necessari a mettere in atto la metodologia proposta ed abbiamo scelto un metodo di dictionary learning (NMF con vincoli di sparsità) adatto all’implementazione dell’algoritmo proposto. Nella presente sezione definiamo l’effettiva implementazione della tecnica e discutiamo il perché delle scelte effettuate. Innanzitutto, il listato 3.3 mostra l’algoritmo nel suo complesso.

Algoritmo 3.3: Interpretation Maximization

```

1: function IM(dataset,model, $\lambda_{DISTANCE}$ ,  $\lambda_1$ ,  $\lambda_2$ , numAtoms, maxIterDL,
   maxIterIM,i,k1,k2)
2:    $W \leftarrow NMFSC(dataset, \lambda_1, \lambda_2, numAtoms, maxIterDL)$ 
3:    $prob\_i \leftarrow model.inference(i)$ 
4:    $prob\_i \leftarrow oneHot(argMax(prob\_i))$ 
5:    $h \leftarrow random(numAtoms, 1)$ 
6:    $iterCount \leftarrow 0$ 
7:   repeat
8:      $x \leftarrow W \times h$ 
9:      $prob\_x \leftarrow model.inference(x)$ 
10:     $loss \leftarrow softmaxCrossEntropy(prob\_i, prob\_x) + \lambda_{DISTANCE} \times \|x - i\|^2$ 
11:     $h \leftarrow ADAMOptimizer(loss, 0.01)$ 
12:     $h \leftarrow proj(h, k1, k2)$ 
13:     $iterCount \leftarrow iterCount + 1$ 
14:   until convergence  $\vee iterCount > maxIter$ 
15:   return  $h$ 
16: end function

```

La prima operazione necessaria è quella del calcolo del dizionario. Quest’ultimo è ottenuto tramite l’uso di NMF con vincoli di sparsità, descritto precedentemente in *Sezione 3.2.2*. Per essere concisi, questa operazione è stata integrata con il resto dell’algoritmo, è preferibile tuttavia precalcolare il dizionario e poi caricarlo al momento della generazione della spiegazione. In seguito, riga 3, si carica il modello e si effettua un’inferenza sull’immagine in input in modo da ottenere un vettore di probabilità $prob_i$, con una voce per ogni classe. Per l’uti-

lizzo all'interno della loss modifichiamo il vettore delle probabilità in un vettore one-hot, ossia un array con tutti 0 ed un uno al posto del massimo valore presente nella struttura. A questo punto, generiamo un encoding random di dimensioni $numAtoms \times 1$, questo sarà il punto di partenza dell'algoritmo. Dalla riga 7 inizia il processo di ottimizzazione, che termina una volta raggiunta la convergenza o dopo un massimo numero di iterazioni prestabilite. Il primo step del processo, prevede il calcolo di un'immagine sintetica x data dal prodotto tra il dizionario W e l'encoding h , ossia da una combinazione lineare di atomi del dizionario. Successivamente, si effettua un'inferenza sull'immagine sintetica, in modo da ottenere un vettore di probabilità $prob_x$. A questo punto si calcola la loss, data dalla cross entropy, $-\sum_k prob_i_k \times \log(prob_x_k)$, e da una misura di distanza tra l'immagine sintetica e quella reale. La minimizzazione della loss avviene tramite discesa del gradiente, per cui è possibile sfruttare qualsiasi ottimizzatore si abbia a disposizione, la nostra scelta ricade su ADAM (ADAptive Moment-estimation) [Kingma and Ba, 2014]. Al termine dell'operazione di discesa del gradiente l'encoding risultante, oltre a non risultare sparso, potrebbe non essere completamente positivo. Per riportarlo in una situazione ideale, sfruttiamo l'operatore di proiezione proposto da [Hoyer, 2004].

A questo punto, la prima iterazione del processo di ottimizzazione termina e si ripetono le operazioni appena descritte fino al raggiungimento della convergenza o del massimo numero di iterazioni. Una volta terminato il processo viene restituito in output l'encoding che consente di selezionare gli atomi più rilevanti in merito alla quantità di interesse utilizzata. Nel prossimo capitolo sarà testata l'efficacia della tecnica proposta.

Capitolo 4

Test e Risultati

Dopo aver esaminato i concetti di base relativi al problema dell’interpretabilità, aver analizzato i metodi principali per affrontare tale questione, aver proposto un nuovo approccio innovativo al problema dell’interpretabilità, passiamo ora ai test e all’analisi dei risultati. Nel corso del presente capitolo, analizzeremo in modo minuzioso i dataset su cui basare i test, cercheremo quindi di esaminarne le caratteristiche ed il motivo del loro utilizzo. Passeremo poi allo studio del modello utilizzato per testare la tecnica: LeNet-5, una rete convoluzionale messa a punto da [Lecun et al., 1998]. A questo punto metteremo sotto stress la tecnica di dictionary learning utilizzata, NMF con vincoli di sparsità, cercando di comprendere quale sia la miglior combinazione di parametri necessari alla generazione di un dizionario conforme alle esigenze della tecnica proposta. Infine, testeremo in modo approfondito la metodologia esposta nel *Capitolo 3*, sia dal punto di vista dell’interpretabilità locale che da quello dell’interpretabilità globale. Cercando di comprendere quali siano i punti di forza e di debolezza del metodo messo in atto. Ricordiamo infine che, essendoci posti nel contesto della post-hoc interpretability, i test saranno effettuati su modelli preaddestrati.

4.1 Descrizione dei dataset

Nelle prossime sezioni descriveremo la struttura e le caratteristiche fondamentali dei dataset su cui andremo ad effettuare i test per la tecnica proposta. Analiz-

zeremo dapprima MNIST, un dataset di cifre scritte a mano e successivamente Fashion-MNIST, un dataset di capi di abbigliamento sviluppato come rimpiazzo di MNIST.

4.1.1 MNIST

MNIST è un database di cifre scritte a mano, messo a punto da Yann LeCun, Corinna Cortes e Christopher J.C. Burges [LeCun and Cortes, 2010]. Il dataset in questione è costituito da un trainig set di 60.000 elementi ed un test set di 10.000 elementi. È per lo più un sottoinsieme del più vasto NIST¹ (in particolare NIST Special Database-3 e NIST Special Database-1) sul quale sono state effettuate operazioni di preprocessing per centrare le immagini sulla base del loro centro di massa e fare in modo che siano tutte della stessa dimensione. Il training set di MNIST è composto da 30.000 pattern estratti da SD-3 e da 30.000 elementi prelevati da SD-1. Le cifre presenti nel trainig set sono state create da 250 individui. Allo stesso modo il test set è composto da 5.000 elementi estratti da SD-3 e da 5.000 pattern presi da SD-1. Le cifre presenti nel test set sono state messe a punto da 250 individui. È importante notare che l'insieme degli individui che ha scritto le cifre per il training set è disgiunto da quello delle persone che hanno scritto le cifre per il test set. La *Figura 4.1* mostra alcune immagini estratte dal dataset.



Figura 4.1: Esempi di immagini estratte dal dataset MNIST

¹<https://www.nist.gov/srd/nist-special-database-19>, Ultimo accesso in data 20/09/2018

Vale, inoltre, la pena sottolineare che le immagini originali di NIST erano in bianco e nero (bilevel). Sono state normalizzate e ridimensionate per entrare in una griglia di 20x20 pixel mantenendone l'aspect ratio. Il risultato di questa operazione ha prodotto delle immagini grayscale² che sono state centrate, sulla base del centro di massa, in una griglia di 28x28 pixel.

4.1.2 Fashion-MNIST

Fashion-MNIST è un dataset messo a disposizione da Zalando [Xiao et al., 2017], che si propone come alternativa più complessa a MNIST. In particolare, questo dataset eredita tutte le caratteristiche più importanti di MNIST, riuscendo però ad aumentare la complessità del task di classificazione. Entrando nello specifico Fashion-MNIST è un dataset di immagini di abbigliamento, costituito da un training set di 60.000 elementi ed un test set di 10.000 elementi. Ogni punto dello spazio vettoriale in questione è un'immagine grayscale 28x28 cui è associata una label, da 0 a 9, che indica la categoria di appartenenza dell'indumento.

Label	Descrizione
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

²Sono immagini in cui il valore di ogni pixel rappresenta solo una quantità di luce, o intensità. [Johnson, 2006] Questo valore è espresso va da un minimo di 0, colore nero, ad un massimo di 1, colore bianco. I valori intermedi rappresentano le sfumature di grigio. Per rappresentare tali immagini si utilizzano 8 bit per pixel. In questo modo si possono visualizzare 256 livelli di grigio.

Proprio per le caratteristiche esposte, Fashion-MNIST è stato sviluppato con l'idea di essere un rimpiazzo rapido e, allo stesso tempo, più impegnativo di MNIST. La *Figura 4.2* mostra alcuni esempi di immagini estratte dal dataset.



Figura 4.2: Esempi di immagini estratte dal dataset Fashion-MNIST

Spesso in letteratura si trovano giudizi contrastanti su MNIST. Molti ritengono che se una tecnica sviluppata non funziona su MNIST allora non funziona affatto. In tal senso, questo dataset rimane una pietra miliare del Machine Learning. Tuttavia, c'è anche da dire che se una tecnica ha funzionato su MNIST potrebbe fallire su dataset più complessi. Spesso, infatti, MNIST è ritenuto dagli esperti del settore un dataset ormai troppo semplice, sul quale le reti convoluzionali raggiungono un'accuratezza del 99.7% e anche algoritmi di Machine Learning più classici riescono ad ottenere prestazioni estremamente elevate. Dunque, per questi motivi è stato proposto Fashion-MNIST, che consente di mantenere tutti i vantaggi di MNIST, dovuti in larga parte al preprocessing delle immagini, aumentando la complessità del task in questione.

4.2 Descrizione del sistema di classificazione utilizzato: LeNet-5

LeNet-5 è una rete neurale di tipo convoluzionale (CNN) progettata da Yann LeCun per il riconoscimento di caratteri scritti a mano, o stampati [Lecun et al., 1998]. Tale modello può raggiungere un test error rate dello 0.95% su MNIST³. La struttura di questa rete è particolarmente complessa, può essere macroscopicamente scissa in 7 livelli, senza contare lo strato di input, tutti contenenti parametri addestrabili (pesi). L'input è un'immagine 32×32 , decisamente più grande delle

³<http://yann.lecun.com/exdb/mnist/>

cifre presenti nel database, le quali occupano uno spazio di 20×20 in una griglia 28×28 . Questa scelta è stata operata per fare in modo che le caratteristiche fondamentali (corner, edge, etc.) fossero al centro del primo filtro convoluzionale della rete [Lecun et al., 1998].

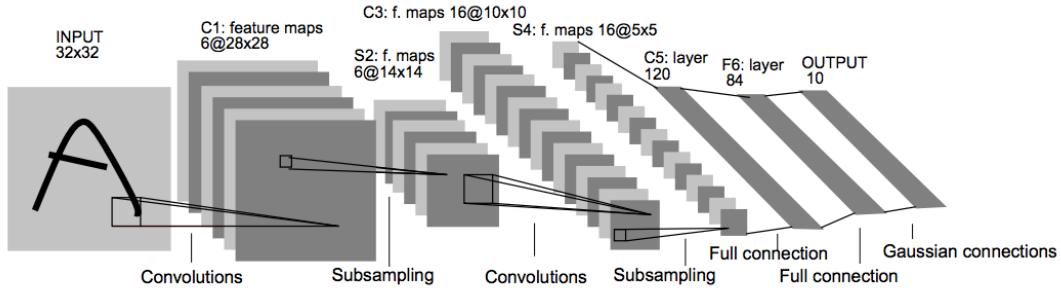


Figura 4.3: Immagine tratta da [Lecun et al., 1998]. In figura è mostrato uno schema riassuntivo dell'architettura di LeNet-5.

Entrando nello specifico, com'è possibile dedurre anche dalla *Figura 4.3* i livelli sono distribuiti nel seguente modo:

- Livello C1, è un livello convoluzionale con 6 feature map. Ogni unità della feature map è connessa all'input mediante un intorno di dimensioni 5×5 . Ogni feature map ha dimensione 28×28 . In totale il livello C1 è composto da 156 parametri e 122.304 connessioni.
- Livello S2, è un livello di subsampling con 6 feature map di dimensione 14×14 . Ogni unità della feature map è collegata al livello C1 mediante un intorno di dimensioni 2×2 . Il livello S2 ha 12 parametri addestrabili e 5.880 connessioni.
- Livello C3, è un livello convoluzionale con 16 feature map. Ogni unità della feature map è connessa, mediante intorni di dimensioni 5×5 , secondo uno schema preciso alle feature map del livello S2. Il livello C3 ha 1.516 parametri addestrabili e 151.600 connessioni
- Livello S4, è un livello di subsampling con 16 feature map di dimensione 5×5 . Ogni unità della feature map è collegata al livello C3 mediante un intorno di dimensioni 2×2 . È costituito da 32 parametri e 2.000 connessioni

- Livello C5 è l’ultimo livello convoluzionale della rete. È composto da 120 feature map. Ogni unità della feature map è collegata a tutte le 16 feature map del livello s4 mediante un intorno di dimensioni 5×5 . La dimensione delle feature map in C5 è di 1×1 . Il livello C5 ha 48.120 connessioni addestrabili.
- Livello FC6 è full connected ed è costituito da 84 unità. Ha 10.164 parametri addestrabili.
- Livello FC7 è l’output della rete, costituito da 10 unità.

Come si può comprendere dall’analisi dell’architettura del modello, nonostante esso sia stato superato da sistemi più performanti e complessi, rappresenta comunque un ottimo punto di partenza per testare profondamente la tecnica proposta. Il sistema di classificazione⁴ appena esposto sarà utilizzato per i test su entrambi i dataset, MNIST e Fashion-MNIST. A causa della mancanza di modelli preaddestrati per i dataset selezionati, è stato necessario addestrare il LeNet-5, con i seguenti risultati:

- Un’accuratezza del 98.86% sul test set di MNIST.
- Un’accuratezza del 91.43% sul test set Fashion-MNIST.

Sottolineiamo che, sebbene sia stato necessario addestrare il modello, tale fase non fa parte in alcun modo della tecnica proposta. Quest’ultima infatti risulta essere model-agnostic ed orientata all’interpretabilità post-hoc, ossia all’interpretabilità di modelli già addestrati.

⁴Per una descrizione del concetto di classificazione fare riferimento al paragrafo 1.6, *Problemi di Classificazione*

4.3 Test della fase 1: generazione dei dizionari

I dizionari saranno generati con la tecnica studiata nel *Capitolo 3*, NMF con vincoli di sparsità. Come già affermato in precedenza, questa tecnica consente di estrarre un dizionario di atomi particolarmente rilevanti ai fini dell’interpretabilità. La bontà del dizionario generato, però, dipende da una serie di parametri, che vanno studiati e sui quali va fatto fine tuning, in modo da ottenere risultati rilevanti. I parametri che vanno testati sono due:

- La sparsità imposta sul dizionario, che chiameremo λ_1
- La sparsità imposta sull’encoding, che chiameremo λ_2

[Hoyer, 2004] afferma chiaramente che i valori di entrambi i λ , che consentono di ottenere delle buone rappresentazioni, sono nell’intervallo [0.6, 0.8]. Pertanto, nei test effettuati, andremo ad analizzare le combinazioni dei due parametri nell’intervallo [0.6, 0.9]. Questo approccio ci consentirà di comprendere meglio come effettuare la scelta del dizionario, oltre ad evidenziare i limiti del metodo messo in atto.

Altro fattore da prendere in considerazione è la porzione di dataset su cui generare il dizionario. In tal senso, individuiamo due possibili opzioni, tra le tante a disposizione. Dapprima generiamo dei dizionari prendendo in considerazione il dataset nella sua completezza. Una volta effettuata un’analisi approfondita dei dizionari generati, passiamo alla produzione di dizionari sulle singole classi. Lo studio di questi due casi notevoli ci farà comprendere meglio quale sia la scelta più opportuna del dizionario a seconda del contesto in cui operiamo.

4.3.1 Test dei dizionari su MNIST

Ci concentreremo in questa sezione sul problema della generazione dei dizionari relativi al dataset MNIST, esplorato in *Sezione 4.1.1*.

Sebbene le tecniche di dictionay learning puntino alla generazione di un dizionario overcomplete, in cui il numero di atomi è molto superiore alla dimensionalità

dell’immagine, le caratteristiche del dataset ci consentono di impostare un ragionamento più efficiente. Infatti, i dati presenti in MNIST sono estremamente sparsi, quindi, è presumibile che ci sia una dimensionalità effettiva decisamente minore della dimensionalità del input. Si può, così, scegliere un numero di atomi maggiore o paragonabile alla dimensionalità intrinseca e, contemporaneamente, molto minore della dimensionalità del input. Come scegliere quindi questo valore? In casi come questo la letteratura suggerisce di applicare la Principal Component Analysis (PCA) [F.R.S., 1901] e scegliere un numero di componenti in grado di catturare circa il 95% della varianza dei dati e selezionare un numero di atomi pari al doppio del valore restituito dalla PCA. Nel caso di MNIST, seguendo questo ragionamento, saranno utilizzati dizionari con 200 atomi.

Di seguito valutiamo i dizionari generati **sull’intero dataset**, per i quali facciamo variare i valori di λ_1 e λ_2 , esplorandone l’effetto delle varie combinazioni. La *Figura 4.7* si propone come schema riassuntivo a supporto della scelta del dizionario migliore.

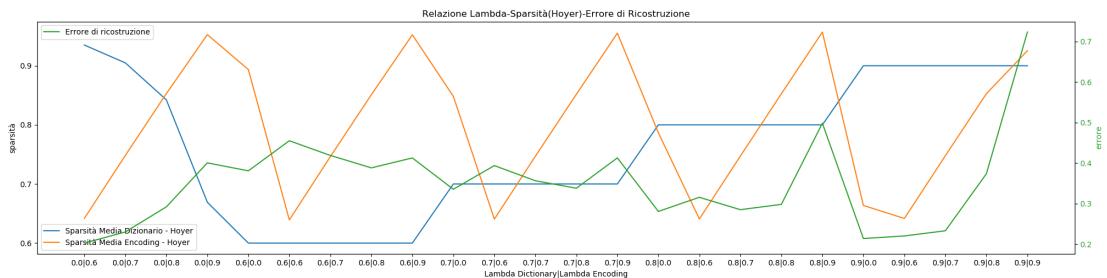


Figura 4.4: Dizionari generati mediante NMF con vincoli di sparsità su MNIST. I dizionari sono generati su tutte le classi del dataset. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell’errore di ricostruzione.

Il grafico in *Figura 4.7* è strutturato nel seguente modo. Sull’asse delle ascisse sono posizionate le varie combinazioni di λ_1 (sparsità imposta sul dizionario) e λ_2 (sparsità imposta sull’encoding) che assumono entrambi i seguenti valori $[0,0.6,0.7,0.8,0.9]$. Sulle ordinate sono posizionati due assi, quello di sinistra rappresenta la scala della sparsità come calcolata in *Sezione 3.2.2*, quello di destra presenta i valori dell’errore di ricostruzione calcolato come norma di Frobenius

tra il dataset e la ricostruzione tramite dizionario ed encoding, ossia $\|V - WH\|_F$. L’immagine consente, quindi, di avere una visione di insieme su tre valori principali, la sparsità effettiva sul dizionario, la sparsità effettiva sull’encoding e l’errore di ricostruzione. Dall’analisi del suddetto grafico si evince che ci sono tre coppie di valori che consentono di avere un buon trade-off tra errore di ricostruzione basso e sparsità su encoding e dizionario alte:

λ_1	λ_2
0	0.8
0.8	0
0.8	0.7

I risultati ottenuti sono in linea con le deduzioni effettuate da [Hoyer, 2004], che propone come valore ottimo su un dataset di volti i seguenti valori $\lambda_1 = 0$ e $\lambda_2 = 0.75$. Infatti, se andiamo a visualizzare gli atomi dei dizionari generati (*Figura 4.5*), notiamo come essi siano chiari e ben strutturati.

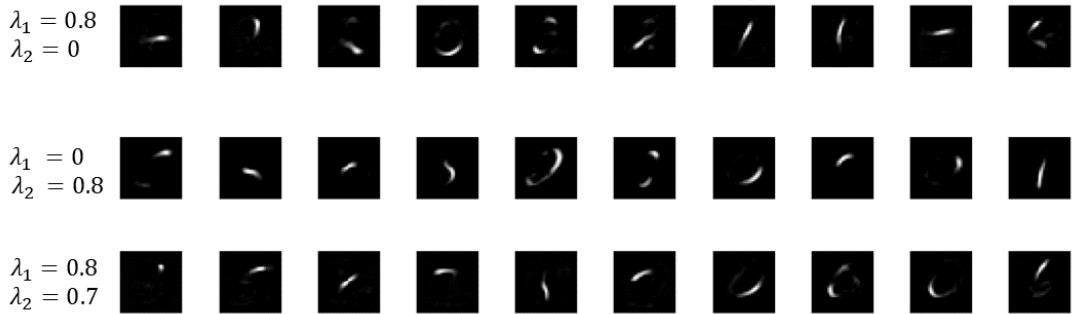


Figura 4.5: Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.

Pertanto i test della tecnica completa proposta saranno basati su uno di questi tre dizionari che esibiscono caratteristiche molto interessanti. Il caso peggiore invece appartiene alla combinazione di valori $\lambda_1 = 0.6$ e $\lambda_2 = 0.6$, per i quali si riscontra un errore di ricostruzione elevato a fronte di una sparsità, nell’encoding e nel dizionario, relativamente bassa. Dalla *Figura 4.6*, che mostra atomi estratti dal suddetto dizionario, si comprende come a valori non ottimali nel grafico in *Figura 4.7* corrispondano atomi con caratteristiche non eccelse per i fini della tecnica proposta. In particolare, vale la pena notare che le rappresentazioni

estratte sono decisamente più globali rispetto a quelle in *Figura 4.5* e spesso un singolo atomo contiene una cifra completa.



Figura 4.6: Esempi di atomi estratti in modo random dal dizionario con i valori di lambda ritenuti peggiori.

Fino a questo momento abbiamo analizzato dizionari generati sull'intero dataset. Un'altra possibile opzione, che velocizzerebbe notevolmente i tempi di elaborazione, sarebbe la generazione di un dizionario apposito per la classe che si intende spiegare. Studiamo, quindi, questa situazione valutando i vantaggi ed eventualmente gli svantaggi che possono nascere da un approccio del genere.

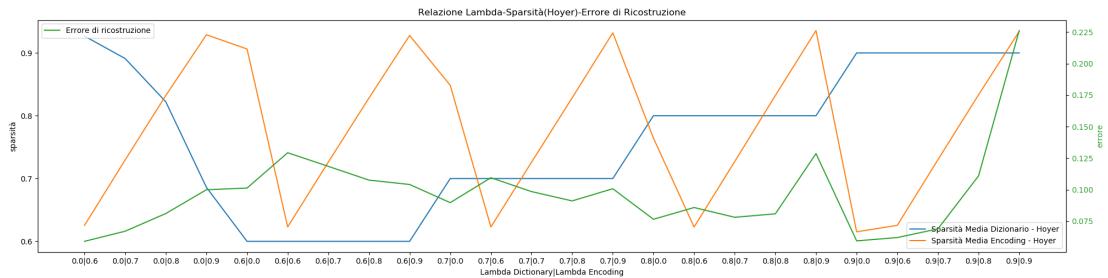


Figura 4.7: Dizionari generati mediante NMF con vincoli di sparsità su MNIST. I dizionari sono generati sulla classe 8. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell'errore di ricostruzione.

I risultati ottenuti sono consistenti con quelli dei dizionari generati su tutte le classi del dataset. Va notato però che in questo caso l'errore di ricostruzione è sempre molto basso. Questo è un indizio del fatto che ricostruire gli elementi di una classe, avendo a disposizione atomi generati in modo apposito per quella classe, facilita molto il task di approssimazione. D'altra parte la scelta di generare un dizionario sulla singola classe ci limita alla possibilità di produrre spiegazioni solo per quella classe. I valori di λ_1 migliori sono quindi:

λ_1	λ_2
0	0.8
0.8	0
0.8	0.7
0.9	0.8

In *Figura 4.8* sono mostrati esempi di atomi estratti dai dizionari con valori di λ_1 e λ_2 ottimi. Le strutture sono ben riconoscibili e significative, ovviamente essendo generate a partire solo dalla classe 8, fanno riferimento alla classe stessa.

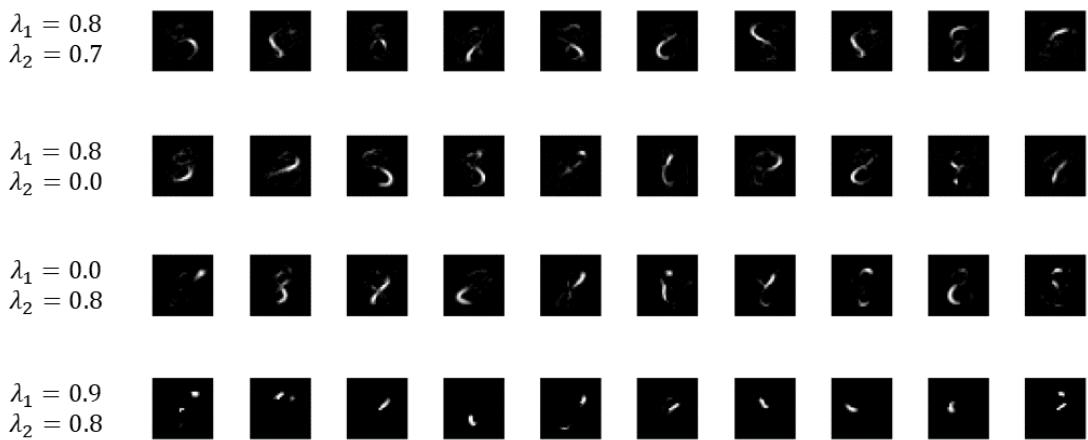


Figura 4.8: Esempi di atomi estratti in modo random dai dizionario con i valori di lambda ritenuti migliori.

Ancora una volta la combinazione peggiore è $\lambda_1 = 0.6$ e $\lambda_2 = 0.6$, in *Figura 4.9*. Infatti, gli atomi estratti da questo dizionario risultano globali e decisamente poco utili ai fini della tecnica proposta.



Figura 4.9: Esempi di atomi estratti in modo random dal dizionario con i valori di lambda ritenuti peggiori.

4.3.2 Test dei dizionari su Fashion-MNIST

Passiamo ora all'analisi dei dizionari generati per il dataset Fashion-MNIST, descritto in modo approfondito in *Sezione 4.1.2*. Dopo alcuni test preliminari, si

è deciso di generare dizionari in modo leggermente differente rispetto a MNIST. Invece di utilizzare tutte le combinazioni di λ_1 e λ_2 a disposizione, sono stati generati dizionari in cui la sparsità viene imposta, in modo mutamente esclusivo, sul dizionario oppure sull'encoding. Infatti, gli atomi prodotti con l'imposizione della sparsità sia sul dizionario sia sull'encoding sono risultati costituiti da artefatti non rilevanti ai fini della tecnica proposta, come constatabile in *Figura 4.10*.



Figura 4.10: Esempi di atomi estratti da un dizionario generato su Fashion-MNIST in cui è stata applicata la sparsità sia sul dizionario che sull'encoding. In particolare $\lambda_1 = 0.8$ e $\lambda_2 = 0.7$. Una delle migliori combinazioni di parametri per MNIST.

Sebbene, l'algoritmo descritto in *Sezione 3.2.2* definisca chiaramente la possibilità di imporre la sparsità contemporaneamente sul dizionario e sull'encoding, sul dataset in questione risultati decisamente migliori si ottengono con l'imposizione solo di una delle due sparsità. Un approccio simile è adottato da [Hoyer, 2004] su un dataset complesso di volti. Il grafico in *Figura 4.14*, riprende lo schema analitico proposto per MNIST e consente di mettere in relazione i valori di λ_1 e λ_2 .

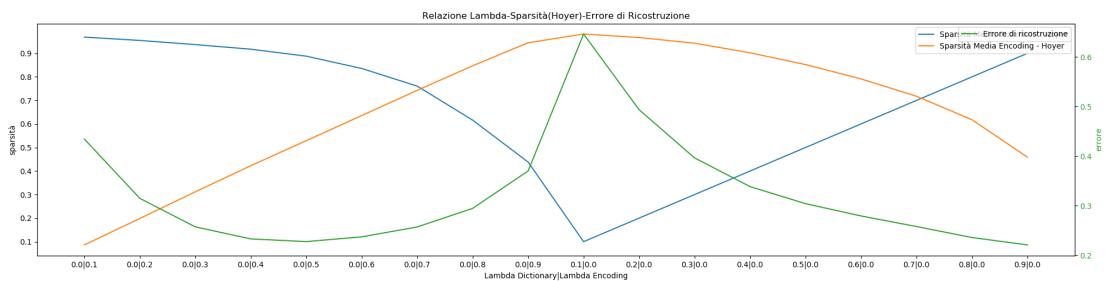


Figura 4.11: Dizionari generati mediante NMF con vincoli di sparsità su Fashion-MNIST. I dizionari sono generati su tutte le classi del dataset. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell'errore di ricostruzione.

Anche in questo caso i risultati ottenuti sono in linea con le deduzioni effettuate da [Hoyer, 2004]. Infatti, se andiamo a visualizzare gli atomi dei dizionari

generati (*Figura 4.12*), notiamo come essi siano chiari e ben strutturati, nonostante l'aumento di complessità delle strutture necessarie a descrivere il dataset in questione. Infatti, tra gli atomi mostrati è possibile individuare facilmente elementi come maniche di pullover, tacchi di scarpe e gambe di pantaloni.

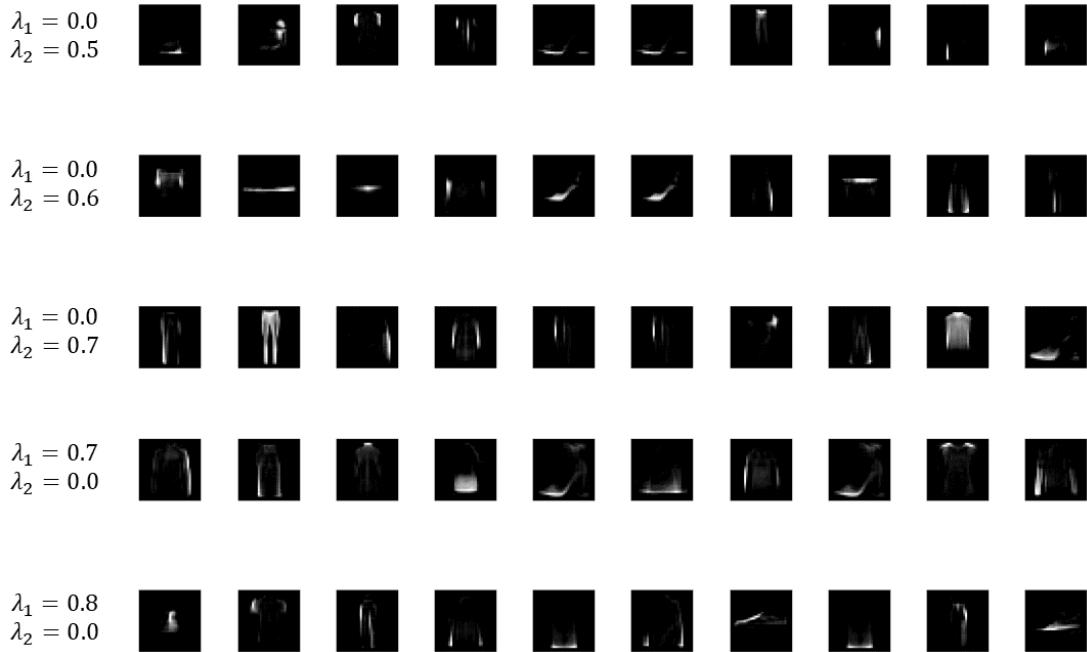


Figura 4.12: Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.

Sulla base dell'analisi del grafico in *Figura 4.14* e dell'analisi visiva degli atomi in *Figura 4.12*, possiamo affermare che i valori di migliori di λ_1 e λ_2 sono:

λ_1	λ_2
0	0.5
0	0.6
0	0.7
0.7	0
0.8	0

Come si evince dall'analisi appena effettuata, non tutti i dizionari generati possono essere utilizzati per il nostro scopo. Alcuni di essi, la maggior parte, non presentano le caratteristiche sufficienti al loro utilizzo nella tecnica completa. In particolare ci sono due situazioni da analizzare:

- Atomi con rappresentazioni globali. Se si impone una sparsità molto alta sull'encoding, lasciando non vincolata quella sul dizionario si ottengono delle rappresentazioni globali, ogni atomo tende a rappresentare un oggetto intero. Lo stesso risultato non vantaggioso si ha imponendo una sparsità bassa sul dizionario, lasciando svincolata quella sull'encoding.
- Atomi eccessivamente sparsi. Tale situazione si verifica nel caso in cui venga imposta una sparsità molto elevata sul dizionario, senza vincolare l'encoding, oppure nel caso in cui sia richiesta una sparsità bassa sull'encoding lasciando svincolata la sparsità sul dizionario.



Figura 4.13: Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.

Passiamo ora all'analisi dei dizionari generati per il dataset Fashion-MNIST, sulla singola classe. Il grafico in *Figura 4.14*, consente, come in precedenza, di mettere in relazione i valori di λ_1 e λ_2 .

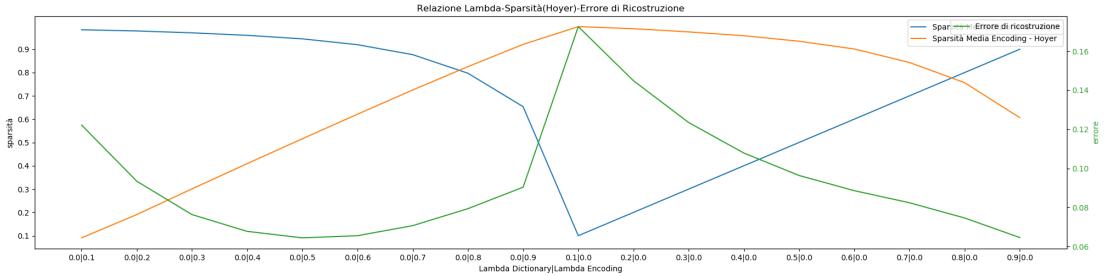


Figura 4.14: Dizionari generati mediante NMF con vincoli di sparsità su Fashion-MNIST. I dizionari sono generati sulla classe 'sandalo'. Sulle ascisse le varie combinazioni di λ_1 e λ_2 . Sulle ordinate i valori di sparsità e dell'errore di ricostruzione.

I risultati ottenuti sono consistenti con quelli dei dizionari generati su tutte le classi del dataset. Va notato però che in questo caso l'errore di ricostruzione è sempre molto basso. Questo è un indizio del fatto che ricostruire gli elementi di una classe, avendo a disposizione atomi generati in modo apposito per quella classe, facilita molto il task di approssimazione. D'altra parte la scelta di generare un dizionario sulla singola classe ci limita alla possibilità di produrre spiegazioni solo per quella classe. I valori di λ_1 migliori sono quindi: λ_2 sono:

λ_1	λ_2
0	0.8
0.8	0

Anche in questo caso i risultati ottenuti sono in linea con le deduzioni effettuate da [Hoyer, 2004]. Infatti, se andiamo a visualizzare gli atomi dei dizionari generati (*Figura 4.12*), notiamo come essi siano chiari e ben strutturati, nonostante l'aumento di complessità delle strutture necessarie a descrivere il dataset in questione. Infatti, tra gli atomi mostrati è possibile individuare facilmente elementi come maniche di pullover, tacchi di scarpe e gambe di pantaloni.

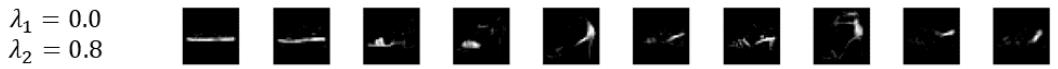


Figura 4.15: Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti migliori.

Nel caso del dizionario generato sulla singola immagine, così come in quello prodotto a partire da tutto il dataset, non tutte le combinazioni di λ_1 e λ_2 riescono ad ottenere dizionari utili. I casi notevoli, in senso negativo, sono sempre gli stessi:

- Atomi con rappresentazioni globali. Se si impone una sparsità molto alta sull'encoding, lasciando non vincolata quella sul dizionario si ottengono delle rappresentazioni globali oppure, al contrario, imponendo una sparsità bassa sul dizionario, lasciando svincolata quella sull'encoding.
- Atomi eccessivamente sparsi. Se si impone una sparsità molto elevata sul dizionario, senza vincolare l'encoding, oppure, nel caso in cui sia richiesta una sparsità bassa sull'encoding, lasciando svincolata la sparsità sul dizionario.



Figura 4.16: Esempi di atomi estratti in modo random dai dizionari con i valori di lambda ritenuti peggiori.

In definitiva, riteniamo che la scelta di generare il dizionario a partire da tutto il dataset o dalla singola classe dipenda dal contesto in cui si sta operando. Su dataset semplici come MNIST, utilizzare un dizionario generato su tutto il dataset potrebbe essere più funzionale. Tuttavia, su dataset più complessi come Fashion-MNIST, sfruttare la generazione di dizionari sulla singola classe potrebbe consentirci di evitare la generazione di dizionari di grandi dimensioni, aumentando la qualità della spiegazione prodotta. Nella sezione successiva, utilizzeremo sia dizionari generati su tutto il dataset sia sulla singola classe per creare delle spiegazioni capaci di aumentare l'interpretabilità locale e globale del modello.

4.4 Test della fase 2: generazione della spiegazione

Nella presente sezione sarà testata in modo approfondito la seconda fase della tecnica, come esposta in *Sezione 3.1.2*. Ci si concentrerà innanzitutto sulla sua applicazione nel contesto dell’interpretabilità locale e successivamente se ne analizzerà l’utilizzo nell’ambito dell’interpretabilità globale. In entrambi i casi, sarà testato il comportamento della tecnica al variare del coefficiente di sparsità descritto nella quantità di interesse 3.2 che riportiamo di seguito per completezza.

$$\max_h \log p(\omega_c | Wh) - \lambda_1 \|Wh - i\|_2 + \lambda_2 \text{sparsity}(h)$$

Quindi, faremo variare λ_2 nell’intervallo [0.1,0.9], mentre λ_1 avrà valore fissato, pari ad 1. Ciò ci consentirà di comprendere meglio l’impatto che ha l’imposizione della sparsità sull’encoding generato.

4.4.1 Test sull’interpretabilità locale

Interpretabilità locale su MNIST

I primi test della tecnica completa saranno effettuati su MNIST, utilizzando un dizionario generato su tutto il dataset con sparsità sul dizionario pari a 0.8 e sparsità sull’encoding pari a 0.7. Supponiamo di voler fornire una spiegazione per l’immagine in *Figura 4.17*.



Figura 4.17: Immagine di un 3 da spiegare.

Per prima cosa otteniamo la probabilità di classificazione del modello nei confronti dell’immagine selezionata. Poi iniziamo il processo di ottimizzazione come descritto in *Sezione 3.1.2*. Ripetiamo il processo per tutti i valori di λ_2 nell’intervallo [0.1,0.9] e valutiamo i risultati ottenuti. I valori migliori, che raggiungono

un buon trade off tra sparsità ed errore di ricostruzione sono sicuramente 0.5,0.6 e 0.7. Per quanto riguarda gli altri, i valori da 0.1 a 0.4, nonostante l'errore di ricostruzione relativamente basso (soprattutto per 0.4), non riescono a fornire spiegazioni adeguate a causa della loro poca sparsità: sono selezionati tanti atomi con un peso molto basso. In *Figura 4.18* sono mostrate le spiegazioni generate.

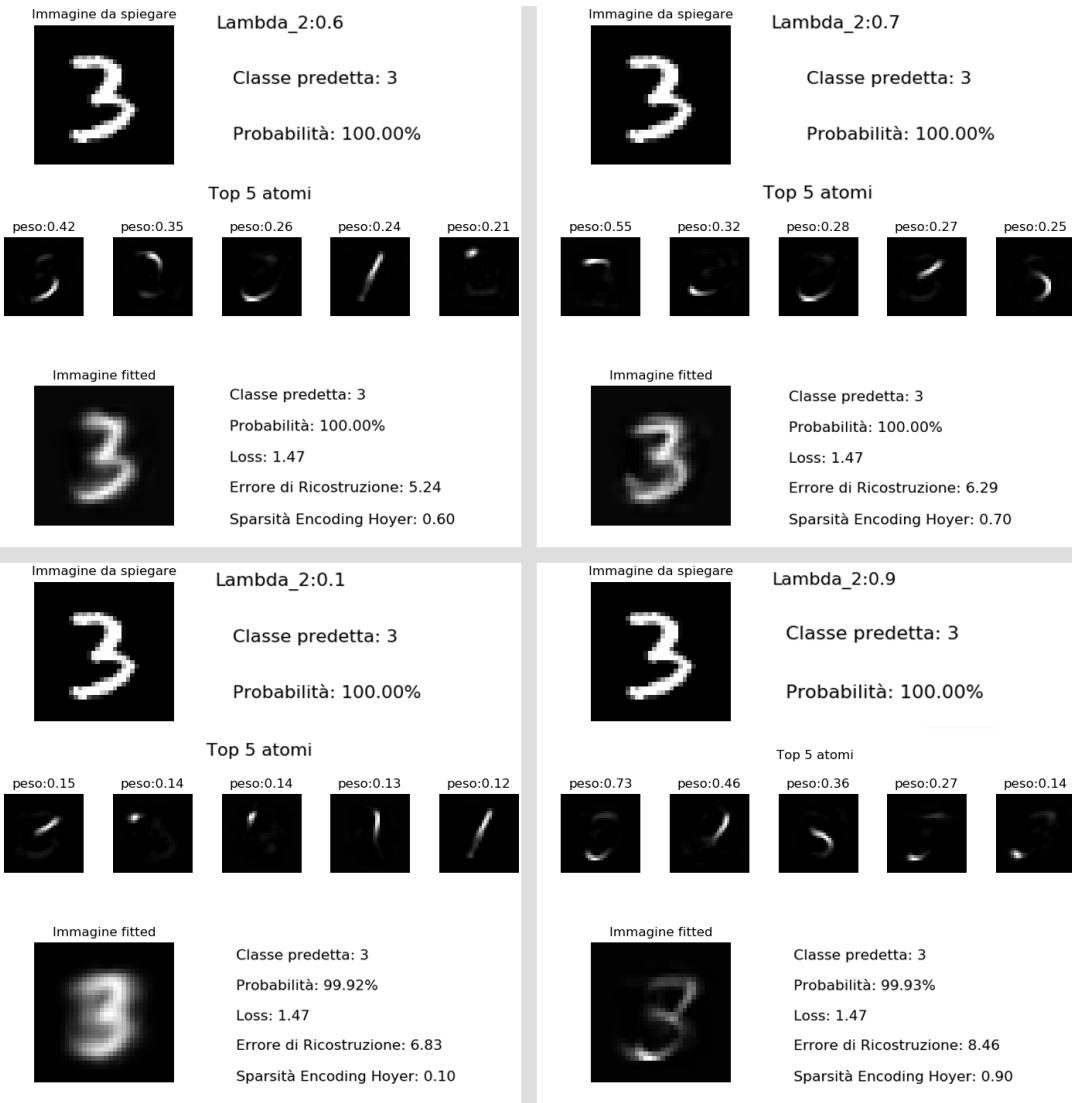


Figura 4.18: Spiegazioni generate per l'immagine 3.

La prima riga dell'immagine contiene rappresentazioni per i valori ottimi di lambda, ossia 0.6 e 0.7. In questi casi, come si può notare, gli atomi selezionati hanno dei pesi rilevanti e consentono al contempo di ottenere una buona ricostruzione dell'immagine originaria. La seconda riga contiene delle spiegazioni meno intuitive. La prima colonna mostra una spiegazione in cui la sparsità imposta è

solo di 0.1. In questo caso gli atomi selezionati hanno un peso eccessivamente basso. Questo è un problema perché se l'encoding seleziona tutti gli atomi del dizionario con un peso molto basso si è in contrasto con il concetto di parsimonia di una spiegazione esposto nel *Capitolo 1*. Di contro, la seconda colonna mostra una situazione diametralmente opposta, in cui viene applicato un valore di sparsità pari a 0.9. In questo caso si ha una pessima ricostruzione dell'immagine (sebbene ricostruire l'immagine non sia il nostro obiettivo, rimane un metro di giudizio, per comprendere la fedeltà della spiegazione all'immagine stessa) e una sparsità eccessiva dell'encoding.

Forniamo di seguito un ulteriore esempio. L'immagine da spiegare è mostrata in *Figura 4.19*. Ancora una volta, i valori migliori, che raggiungono un buon trade off tra sparsità ed errore di ricostruzione, sono sicuramente 0.5 e 0.6, mostrati nella prima riga dell'immagine. In questi casi, come nell'esempio precedente, gli atomi selezionati hanno dei pesi rilevanti e consentono di ottenere una buona ricostruzione dell'immagine originaria.



Figura 4.19: Immagine di un 7 da spiegare.

Anche in questo caso, per prima cosa otteniamo la probabilità di classificazione del modello nei confronti dell'immagine selezionata. Poi iniziamo il processo di ottimizzazione e lo ripetiamo per tutti i valori di λ_2 nell'intervallo [0.1,0.9] valutando così i risultati ottenuti. In *Figura 4.20* sono mostrate le spiegazioni generate.

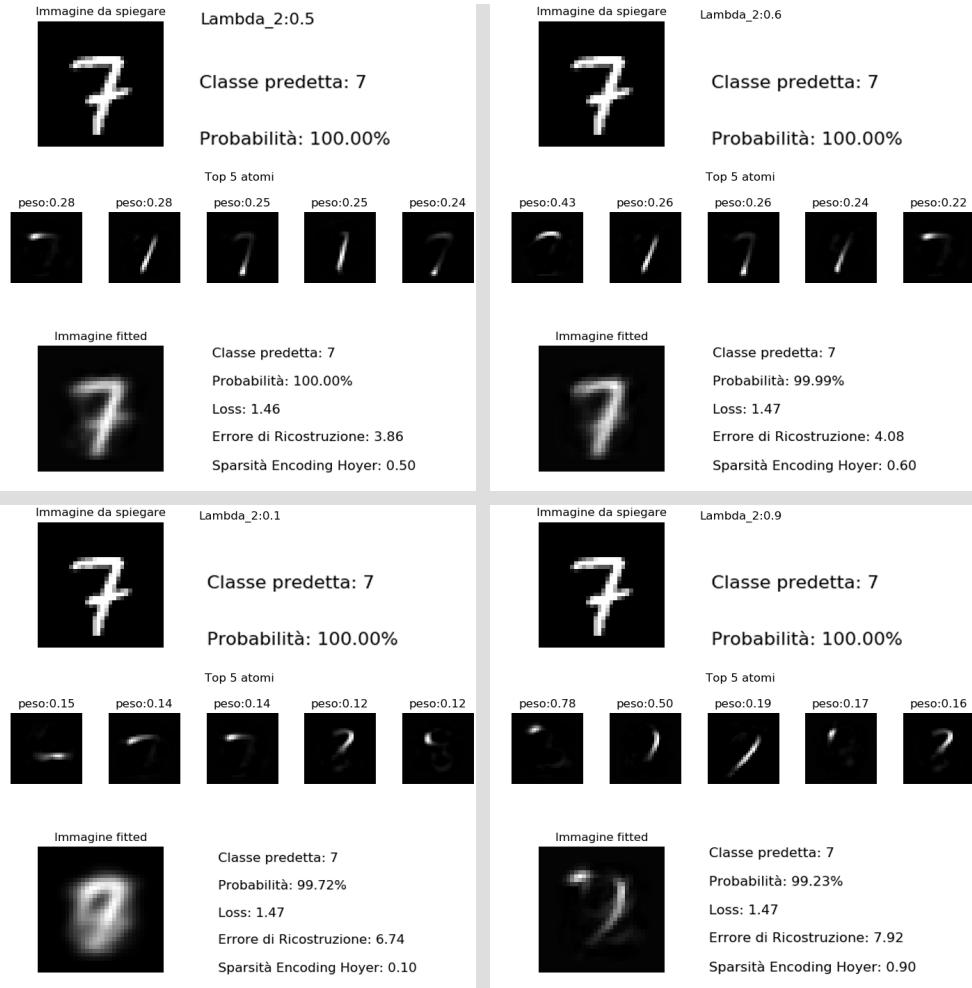


Figura 4.20: Spiegazioni generate per l’immagine 7.

La stessa situazione del caso precedente si ripete anche negli esempi negativi. Infatti, la seconda riga della *Figura 4.20* contiene delle spiegazioni decisamente meno intuitive. La prima colonna mostra una spiegazione in cui la sparsità imposta è solo di 0.1. In questo caso gli atomi selezionati hanno un peso eccessivamente basso. Valgono quindi le riflessioni fatte per il primo esempio. Discorso opposto quando il coefficiente di sparsità assume valore 0.9. In tal caso, si ha un’ottima sparsità dell’encoding associata a valori pessimi per l’errore di ricostruzione. Tra questi due casi è sicuramente preferibile il secondo. Infatti, dal momento in cui il nostro obiettivo non è la ricostruzione dell’immagine di input, bensì la selezione di atomi rilevanti, si può anche soprassedere al fatto che la ricostruzione dell’input non risulti fedele, purché gli atomi selezionati siano significativi.

Testare la variabilità delle spiegazioni

La tecnica proposta nel *Capitolo 3* genera un encoding in grado di selezionare gli atomi più rilevanti per la spiegazione. Per raggiungere questo obiettivo, il metodo parte da un punto random nello spazio vettoriale degli encoding. Siccome la tecnica proposta applica un processo di ottimizzazione mediante ascesa del gradiente non abbiamo la garanzia che essa giunga al massimo globale. Dunque, questo tipo di inizializzazione potrebbe portare alla generazione di una spiegazione ogni volta diversa. Per valutare questa situazione è stata ripetuta la stessa tecnica varie volte sulla stessa immagine. I risultati sono esposti in *Figura 4.21*



Figura 4.21: Generazione della spiegazione 4 volte sulla stessa immagine. Gli atomi selezionati sono leggermente diversi tra loro di volta in volta. Nonostante ciò la spiegazione rimane comprensibile e significativa.

In effetti, come previsto, ogni spiegazione differisce, seppur di poco dalle altre. Ciò potrebbe, in un primo momento, far pensare ad una mancanza di consistenza della tecnica in esame, in realtà ad un'analisi più approfondita si comprende che il risultato ottenuto è decisamente valido. Infatti, la spiegazione non deve necessariamente essere univoca, del resto un essere umano è in grado di fornire diverse spiegazioni dello stesso concetto. Quello che è importante, invece, è la rilevanza degli atomi, che devono essere interpretabili sia come entità a se stanti sia come unione delle parti. Le spiegazioni generate rispettano questo criterio. Per cui, riteniamo che la variabilità delle stesse non vada ad intaccare la validità della tecnica proposta.

Interpretabilità locale su Fashion-MNIST

Riproponiamo in questa sezione la stessa metodologia di test della *Sezione 4.4.1*. Per cui, per prima cosa scegliamo un dizionario. In questo caso utilizzeremo dizionari generati sulle singole classi con sparsità sul dizionario pari a 0.8 e sparsità sull'encoding pari a 0.

Supponiamo di voler fornire una spiegazione per l'immagine in *Figura 4.22*.



Figura 4.22: Immagine di un 'sandal' da spiegare.

Per prima cosa otteniamo la probabilità di classificazione del modello nei confronti dell'immagine selezionata. Poi iniziamo il processo di ottimizzazione come descritto in *Sezione 3.1.2*. Ripetiamo il processo per tutti i valori di λ_2 nell'intervallo $[0.1, 0.9]$ e valutiamo i risultati ottenuti. I valori migliori per la sparsità sull'encoding nella seconda fase della tecnica sono 0.6, 0.7 e 0.9, come riscontrabile in *Figura 4.23*. Valori inferiori a 0.4 non riescono generare spiegazioni adeguate a causa della loro poca sparsità. Vengono, infatti, selezionati tanti atomi con un peso molto basso. In *Figura 4.23* sono mostrate le spiegazioni generate.

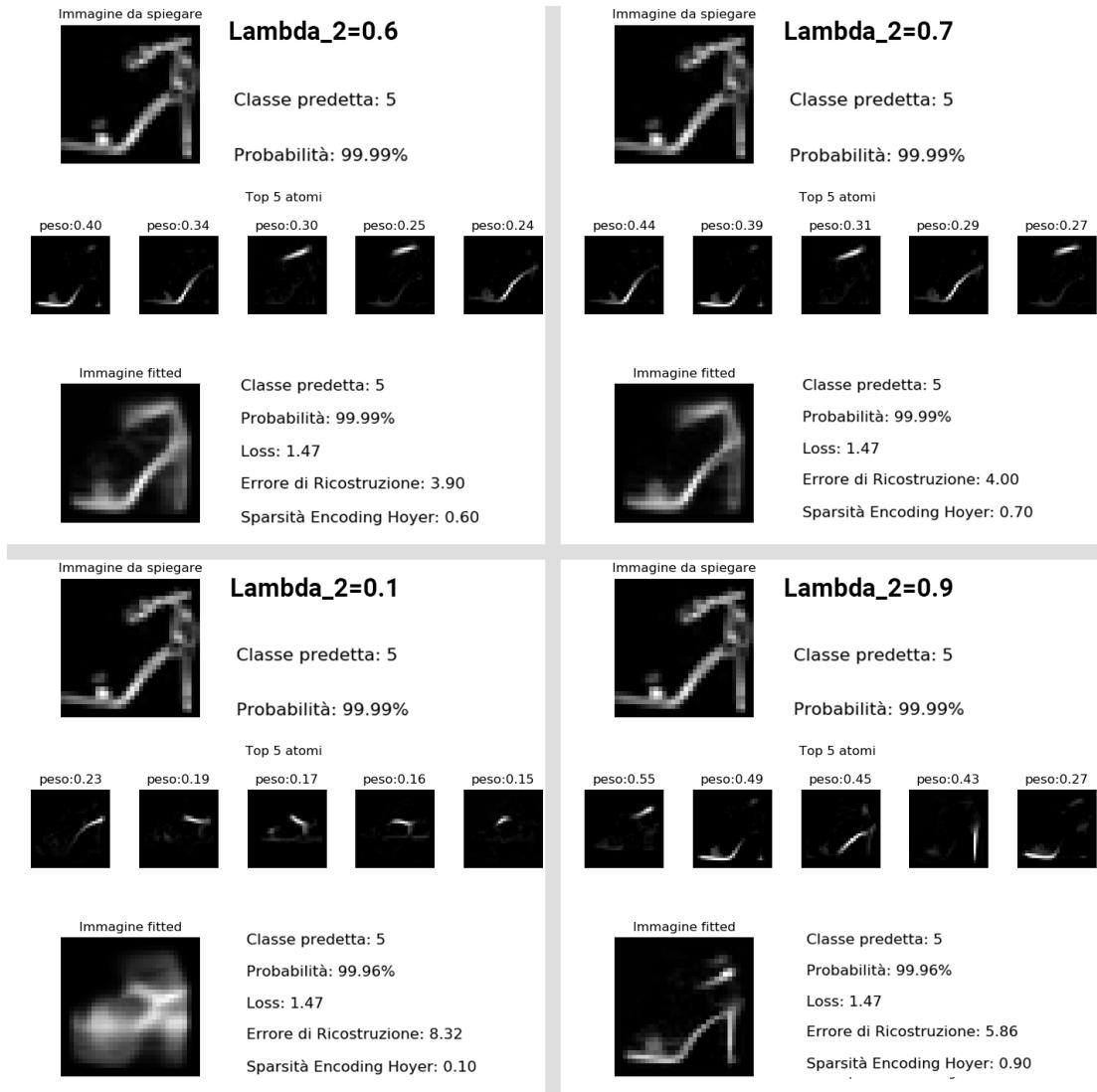


Figura 4.23: Spiegazioni generate per l'immagine 'sandalo'.

Come si può costatare, il valore di λ_2 che genera i risultati peggiori è 0.1, non perchè gli atomi selezionati non siano rilevanti, ma per il loro peso estremamente basso e uniforme tra i vari elementi del dizionario. Si ha una situazione ben diversa con $\lambda_2 = 0.9$, in questo caso si selezionano 4 atomi rilevanti che rappresentano proprio i punti essenziali della scarpa: il tacco, la suola, la punta e la fibbia. Ottimi risultati si riescono ad ottenere anche con valori intermedi del coefficiente di sparsità, 0.6 e 0.7. In questi casi però, si nota come, avendo utilizzato un dizionario su una singola classe, i primi atomi siano leggermente ridondanti.

Forniamo un secondo esempio su Fashion-MNIST. Supponiamo di voler gene-

rare una spiegazione per la classificazione di un pullover, immagine in *Figura 4.24*.



Figura 4.24: Immagine di un 'pullover' da spiegare.

Come in precedenza, per prima cosa otteniamo la probabilità di classificazione del modello nei confronti dell'immagine selezionata. Poi iniziamo il processo di ottimizzazione come descritto in *Sezione 3.1.2* e ripetiamo il processo per tutti i valori di λ_2 nell'intervallo $[0.1, 0.9]$ e valutiamo i risultati ottenuti. I valori migliori, come verificabile in *Figura 4.25*, sono 0.5, 0.6.

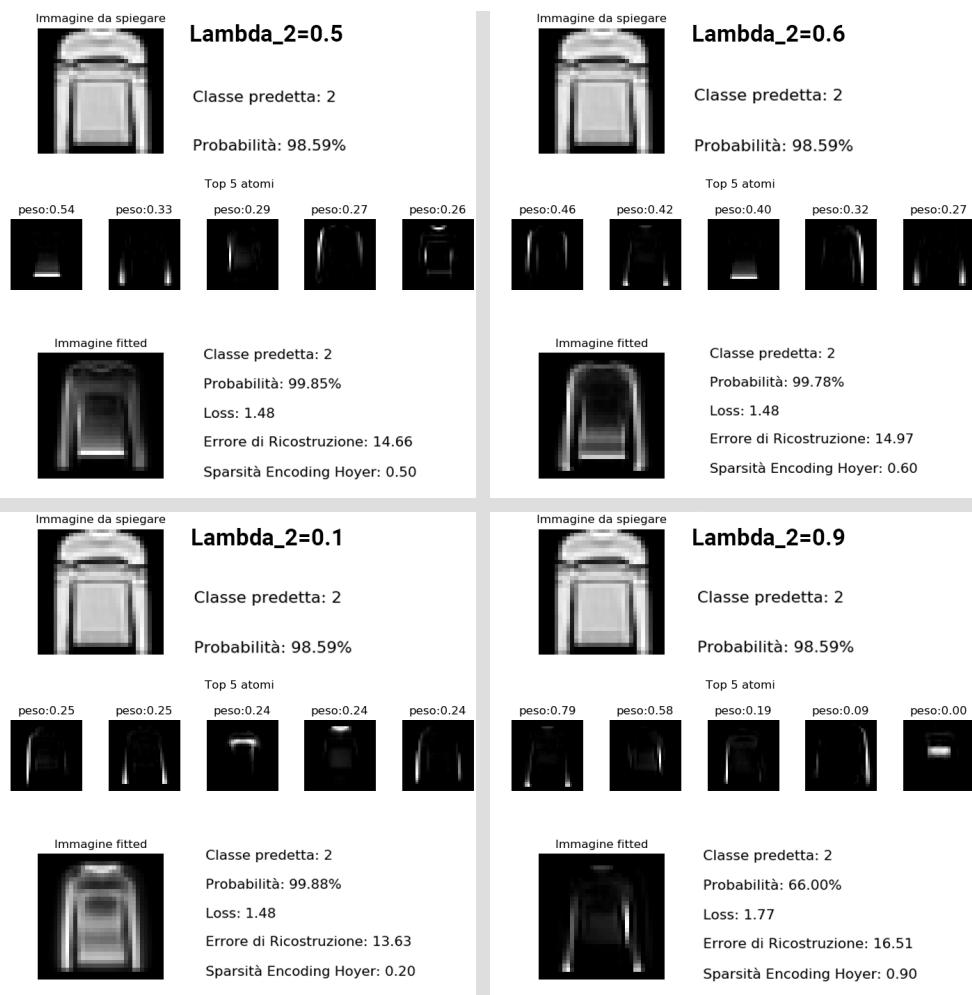


Figura 4.25: Spiegazioni generate per l'immagine 'sandalo'.

Quando il coefficiente della sparsità imposta assume valori inferiori a 0.4, in particolare nell’immagine è visualizzato $\lambda_2 = 0.1$, si hanno situazioni non ottimali. Come nei casi già affrontati gli atomi selezionati hanno tutti un peso molto basso e ciò è in contrasto con i presupposti della tecnica proposta. Per i valori ottimi di λ_2 la tecnica ha un comportamento migliore, in particolare, è possibile notare che le caratteristiche fondamentali che distinguono un maglione sono le maniche e la parte bassa della vita. Riteniamo, invece, che quando il coefficiente della sparsità assume valore pari a 0.9, il numero di atomi selezionati diventa eccessivamente ridotto, in tal caso infatti solo due atomi hanno associati valori dell’econding interessanti, ma ciò non basta a fornire una spiegazione adeguata. Questa situazione è suffragata anche dalla percentuale relativamente bassa con cui viene classificata l’immagine ricostruita.

4.4.2 Test sull'interpretabilità globale

Una volta verificata la validità della tecnica nel caso dell'interpretabilità locale, passiamo al test nel contesto dell'interpretabilità globale. In tal senso, generiamo dei prototipi che ci consentano di comprendere quali sono le strutture più descrittive e caratteristiche per una determinata classe. Anche in questo contesto è bene ricordare che non ci interessiamo alla generazione di un prototipo come immagine nel suo complesso, sfida particolarmente ardua come visto in *Sezione 2.2.1*, ma all'estrazione di patch particolarmente descrittive per una classe specifica. Approfondiamo quindi l'argomento prima su MNIST e successivamente su Fashion-MNIST. In entrambi i casi la quantità di interesse che andiamo ad ottimizzare riprende quella in *Sezione 3.2.2*, che riportiamo per completezza.

$$\max_h \log p(\omega_c | Wh) + \lambda_2 sparsity(h)$$

Ricordiamo che $p(\omega_c | Wh)$ rappresenta la probabilità che l'immagine Wh sia assegnata alla classe ω_c . Il secondo termine, invece, vincola la soluzione ad essere sparsa e non negativa.

Interpretabilità globale su MNIST

Operiamo, dapprima, sul dataset MNIST, per il quale generiamo i prototipi della classe 9, in *Figura 4.26* e della classe 2, *Figura 4.27*. In entrambi i casi generiamo il prototipo per diversi valori di sparsità, imposta sull'encoding generato. In particolare, prendiamo in considerazione gli estremi, 0.1 e 0.9, e dei valori intermedi, 0.6 e 0.7. Questi ultimi forniscono i risultati migliori sia in termini di atomi selezionati che di rilevanza degli stessi. Il valore peggiore, così come per l'interpretabilità locale, rimane 0.1 che, nonostante l'ottima capacità di ricostruzione, non è interessante ai fini della tecnica proposta. Infatti, i valori dell'encoding sono tutti molto bassi e quindi non è ben chiaro quali siano effettivamente gli atomi più caratteristici per la classe in questione.

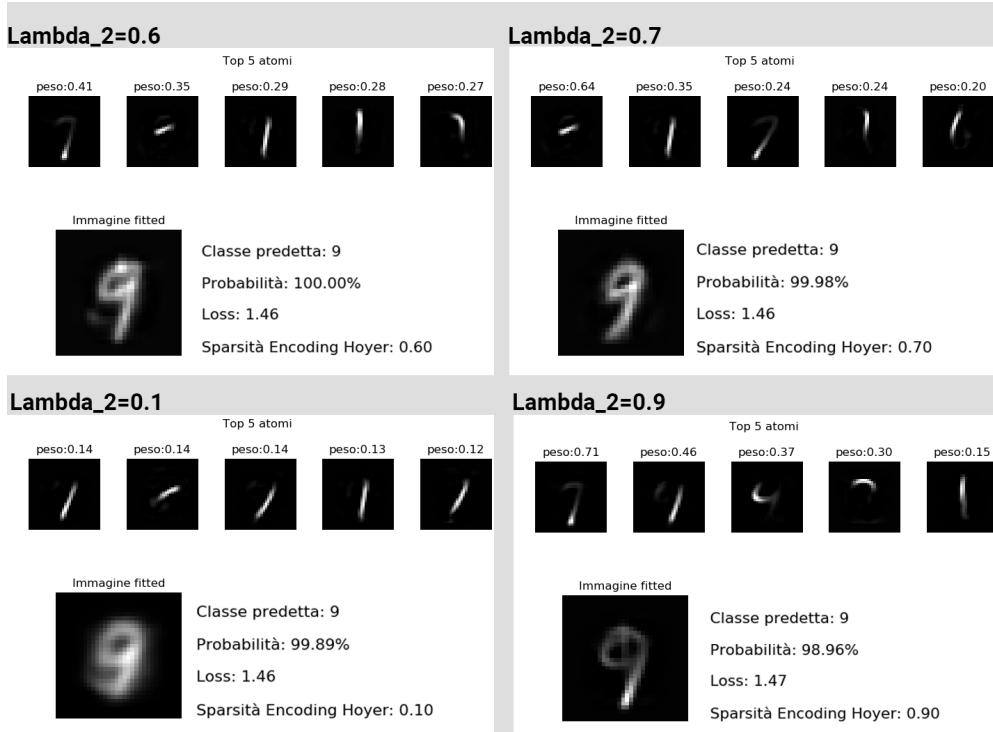


Figura 4.26: Generazione del prototipo per la classe 9. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.

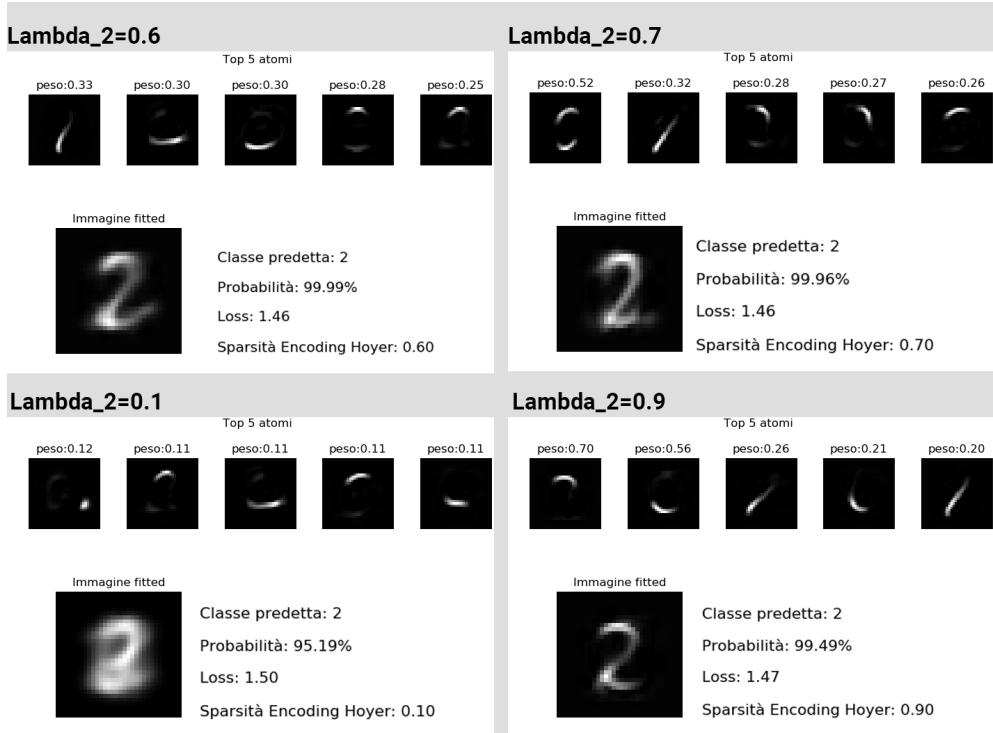


Figura 4.27: Generazione del prototipo per la classe 2. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.

Testare la variabilità dei prototipi su MNIST

Anche nel caso dei prototipi il punto di partenza dell'algoritmo è un encoding random e questo potrebbe portare alla generazione di prototipi sempre diversi. Cerchiamo quindi di comprendere, se effettivamente si verifica questa situazione e valutiamo il suo impatto sull'utilità dei prototipi generati. Come si può notare in *Figura 4.28*, i prototipi differiscono leggermente tra di loro. Valgono tuttavia le osservazioni fatte in *Sezione 4.4.2*, per cui fin quando gli atomi selezionati esprimono caratteristiche tipiche della classe per la quale sono stati selezionati, la tecnica continua ad essere valida.

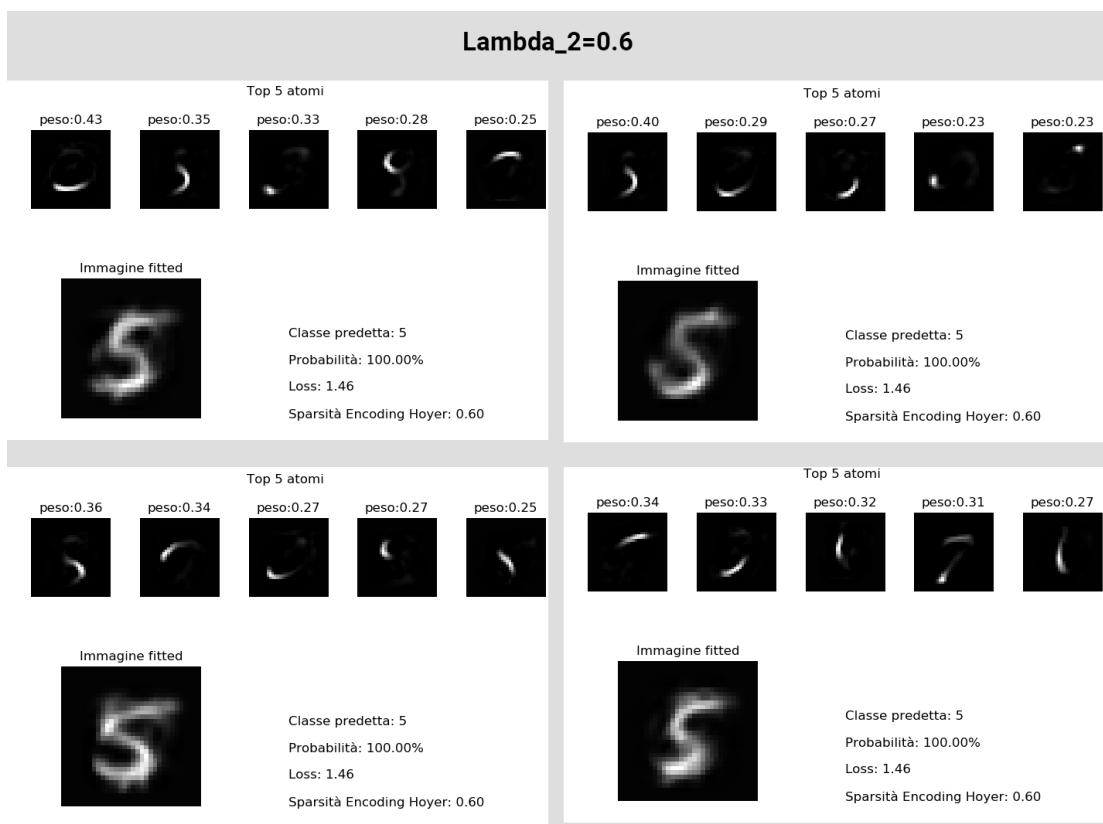


Figura 4.28: Generazione di più prototipi per la classe 5. La variabilità dei prototipi non incide in modo significativo sulla comprensione delle caratteristiche importanti di una classe.

Interpretabilità globale su Fashion-MNIST

Valutiamo ora come la tecnica opera nell'ambito dell'interpretabilità globale, su un dataset più complesso, come Fashion-MNIST. Generiamo quindi i prototipi per le classi 'pullover', in *Figura 4.29*, e 't-shirt', in *Figura 4.30*. Anche in questo

caso prendiamo in considerazione gli estremi, 0.1 e 0.9, e dei valori intermedi, 0.6 e 0.7. Come in precedenza, questi ultimi forniscono i risultati migliori sia in termini di atomi selezionati che di rilevanza degli stessi. Il numero di atomi selezionati con sparsità 0.9 sembra essere eccessivamente ristretto (solo due atomi con un valore significativo). Il valore peggiore è 0.1 che, nonostante l'ottima capacità di ricostruzione, non è interessante ai fini della tecnica proposta. Infatti, i valori dell'encoding sono tutti molto bassi e quindi non è ben chiaro quali siano effettivamente gli atomi più caratteristici per la classe in questione.

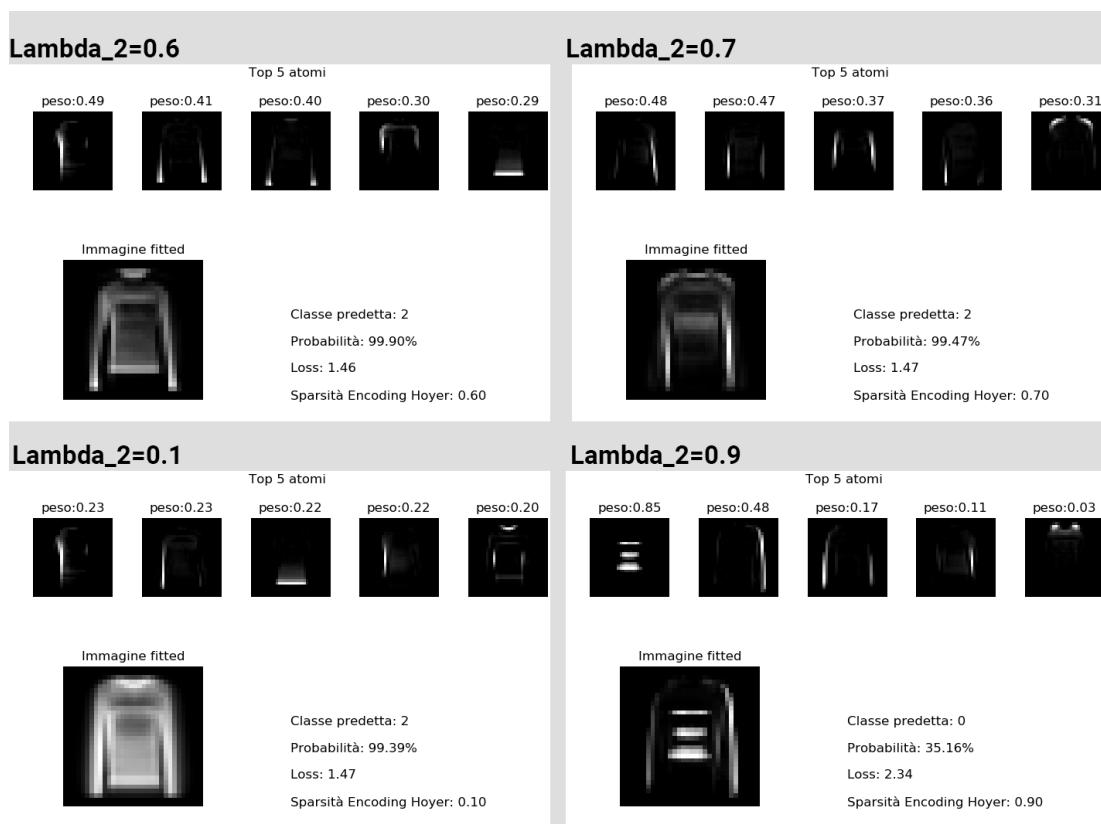


Figura 4.29: Generazione del prototipo per la classe 'pullover'. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.

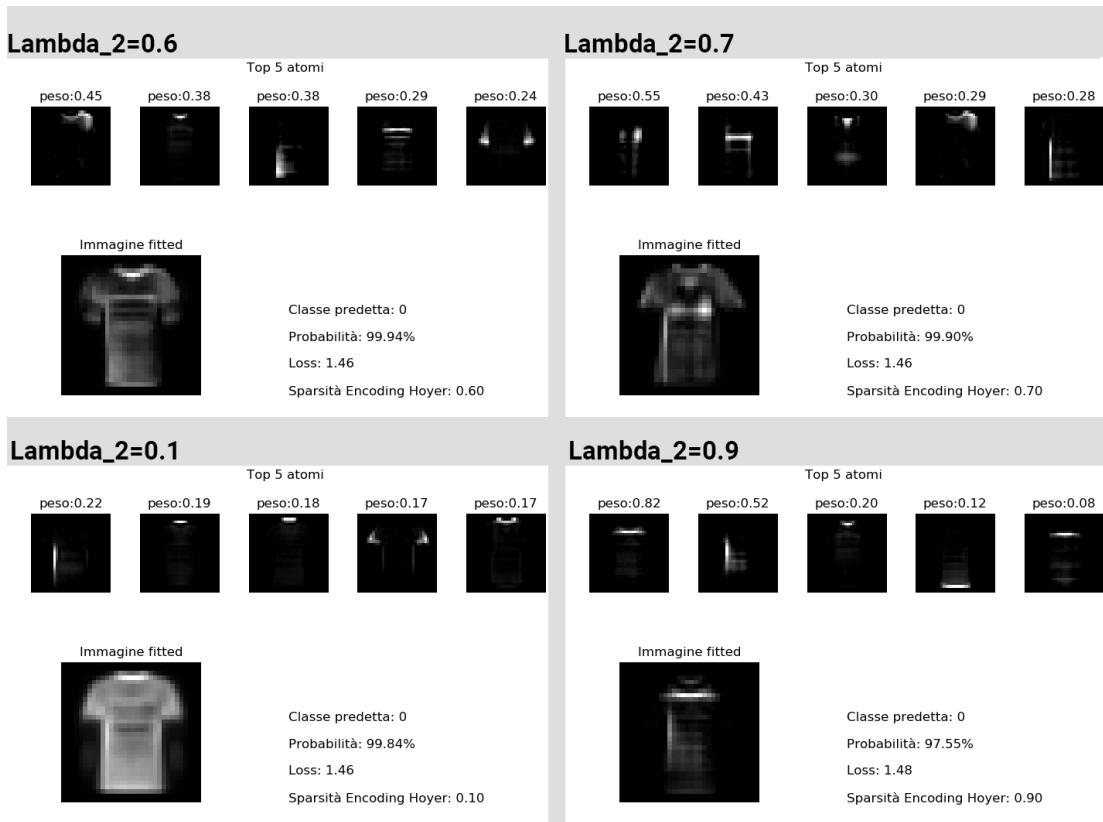


Figura 4.30: Generazione del prototipo per la classe 't-shirt'. I vari livelli di sparsità imposti determinano la bontà del prototipo generato.

Testare la variabilità dei prototipi su Fashion-MNIST

Valutiamo se le affermazioni fatte nel paragrafo precedente possono essere applicate anche quando la tecnica viene trasposta su un dataset più complesso come Fashion-MNIST. Generiamo dunque una serie di prototipi per la classe 'sandallo', in *Figura 4.31*, mantenendo fissi tutti i parametri della tecnica proposta. In questo caso è stato utilizzato un dizionario generato sulla singola classe, 'sandallo' appunto. I prototipi generati sono abbastanza consistenti e, soprattutto, le caratteristiche evidenziate sono il plantare, il tacco e le varie fibie. Tutti tratti caratteristici della classe. Quindi nonostante la variabilità dei prototipi generati essi rimangono comunque consistenti rispetto alle caratteristiche selezionate.

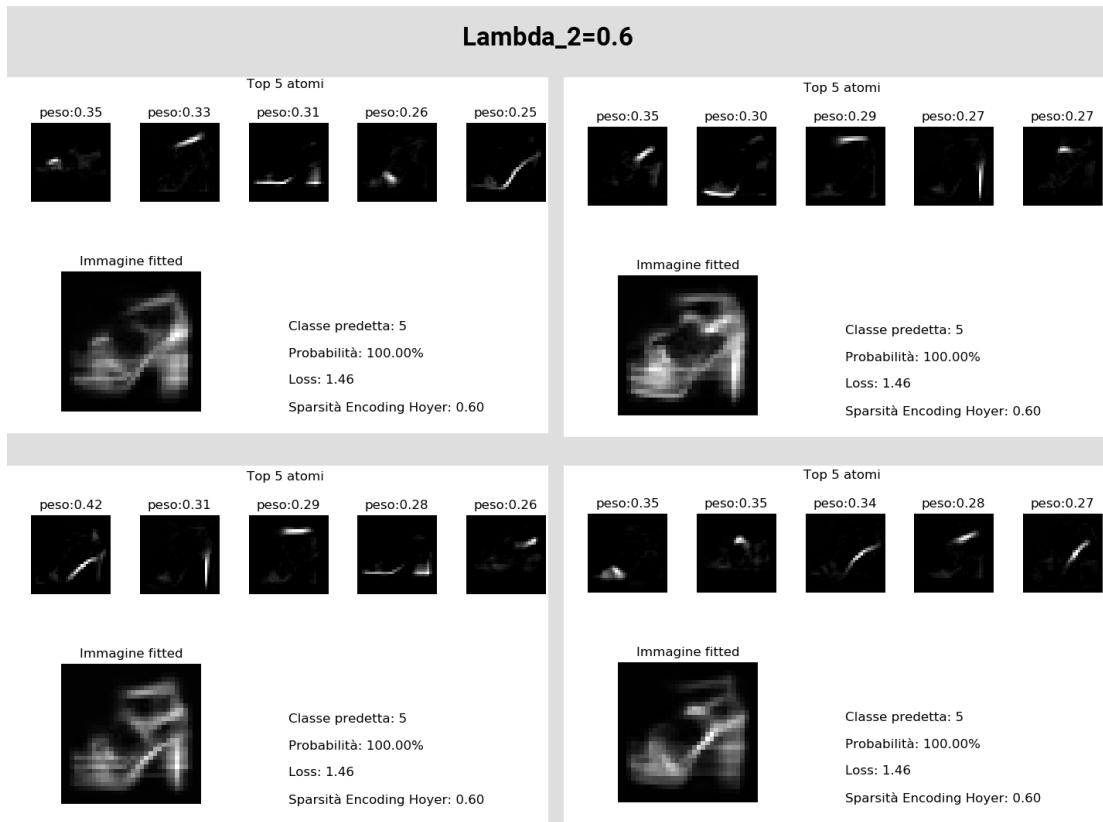


Figura 4.31: Generazione di più prototipi per la classe 'Sandalo'. La variabilità dei prototipi non incide in modo significativo sulla comprensione delle caratteristiche importanti di una classe.

Conclusioni e sviluppi futuri

Lo scopo di questo elaborato di tesi è stato quello di individuare una tecnica che approcciasse il problema dell'interpretabilità dei sistemi di classificazione basati sui Machine Learning da un punto di vista innovativo, facendo leva su tecniche già presenti il letteratura. In particolare, si è deciso di focalizzare il lavoro sullo studio dei sistemi di classificazione delle immagini, contesto in cui riteniamo i metodi attuali non abbiano ancora raggiunto una maturità tale da poter essere utilizzate in produzione. Per fare ciò, è stato necessario un approfondito lavoro di ricerca sull'importanza dei sistemi automatici e sul ruolo che ha il Machine Learning nella loro creazione. Sono stati analizzati i punti interrogativi che l'uso di queste nuove tecnologie ci impone di risolvere. In particolare, si è affrontato il problema dell'interpretabilità dal punto di vista legislativo, analizzando un articolo sull' “Automated individual decision-making, including profiling” presente nel GDPR, un regolamento europeo entrato in vigore il 25 maggio 2018.

A questo punto, si è studiato il concetto di interpretabilità in tutte le sue sfaccettature e si è individuato uno strumento che consentisse di affrontare il problema dell'interpretabilità dei sistemi di Machine Learning complessi: la spiegazione.

Dopo aver discusso sui vantaggi e gli svantaggi nell'uso di tale strumento si è passati all'analisi dello stato dell'arte, con particolare focus su quelle tecniche da cui la nostra metodologia ha tratto maggiore spunto (LIME, AM, Sensitivity analysis). Lo studio di questi algoritmi ci ha consentito di aumentare la nostra competenza in merito al problema dell'interpretabilità e ha posto le basi per la strutturazione della tecnica proposta in questo elaborato. Nel terzo capitolo della presente dissertazione ci si è concentrati sull'effettiva progettazione ed implementazione della metodologia ipotizzata. Si è sviluppato un sistema modulare basato

sulla commistione di tecniche di dicitonary learning e metodologie di ottimizzazione. In particolare, si è fatto uso delle tecniche di dictionary learning per apprendere un dizionario di strutture, patch di immagini, la cui combinazione fosse in grado di generare una spiegazione valida e utile, secondo i criteri esposti nel *Capitolo 1*. Una volta comprese le caratteristiche della tecnica di dictionary learning da utilizzare e scelto tale metodo, si è fatto uso di una tecnica studiata nel *Capitolo 2*, l’Activation Maximization, per selezionare le strutture più caratteristiche nella descrizione di una classe o nella spiegazione di un’immagine. Questo framework, propriamente modificato, ci ha consentito di impostare un procedimento sulla cui base selezionare gli elementi del dizionario per ottimizzare una quantità di interesse, che prendesse in considerazione la probabilità di classificazione dell’immagine generata e la sua vicinanza all’immagine originale. Dopo aver esposto la tecnica dal punto di vista teorico è stato necessario testarla. Gli esperimenti sono stati effettuati su un modello complesso come LeNet-5 e su due dataset MNIST e Fashion-MNIST in modo da comprenderne la validità. In particolare è stato testato l’uso della tecnica sia nel campo dell’interpretabilità locale che in quello dell’interpretabilità globale. Dai risultati ottenuti e valutati nel *Capitolo 4* si è compreso come l’utilizzo di un approccio basato sul dictionary learning sia al contempo il maggior punto di forza e di debolezza della tecnica. L’efficacia di quest’ultima, infatti, dipende in larga parte dalle strutture che si è riusciti ad apprendere nella prima fase. Ad un dizionario con atomi ben progettati, corrispondono spiegazioni decisamente interessanti, che consentono di apprendere aspetti del modello prima sconosciuti. Viceversa, quando il dizionario non è ottimale la qualità della spiegazione cala drasticamente ed in alcuni casi diventa del tutto incomprensibile. In particolare, su dataset leggermente più complicati di MNIST è stato necessario ricorrere ai dizionari generati sulle singole classi per ottenere dei risultati particolarmente validi. Per quanto si sia cercato di testare la tecnica in modo preciso, strutturato e formale, la mancanza di un benchmark per valutare la tecnica è una pecca che ha consentito una validazione dei risultati più qualitativa del dovuto. Si auspica quindi che in futuro sia messo a punto un benchmark per il testing di tecniche nel campo dell’interpretabilità.

Un lavoro preliminare in tal senso è stato svolto da [Mohseni and Ragan, 2018]. Anche in mancanza di un benchmark, per comprendere l'efficacia della tecnica e la possibilità di utilizzarla in un ambiente di produzione, sarà necessario testarla su dataset più grandi (e.g. Imagenet) e modelli più complessi (e.g. Alexnet, GoogleNet, ecc.). Ciò consentirebbe di capire anche la scalabilità del metodo proposto quando si passa da immagini grayscale a immagini a colori. Per sviluppare la tecnica proposta in questo elaborato di tesi, è stato necessario effettuare delle scelte, che ovviamente hanno indirizzato la metodologia in una direzione piuttosto che in un'altra. In tal senso, ci sono molte ramificazioni della tecnica possibili. Una di queste prende in considerazione le spiegazioni per contrapposizione. Infatti, nella dissertazione ci si è concentrati sulla generazione di spiegazioni che possano rispondere a domande del tipo: *“Perché l’immagine è stata classificata dal modello come una scarpa al 99%?”*. Una direzione interessante in cui procedere potrebbe essere quella di rispondere a domande del tipo: *“Perché l’immagine è stata classificata come una scarpa e **non** come uno stivale?”*. Questo tipo di giustificazioni sono alla base del nostro modo di ragionare quotidiano e riteniamo possano svolgere un ruolo importante anche nel campo dell’interpretabilità. Reputiamo, inoltre, necessario individuare e testare la tecnica con altre metodologie di dictionary learning. Infatti, NMF con vincoli di sparsità consente di ottenere dei buoni risultati, ma non dà la possibilità di imporre una struttura agli atomi in modo esplicito. Individuare un nuovo metodo in tal senso potrebbe migliorare le prestazioni della tecnica nel suo complesso. Sarebbe anche molto utile riuscire a risolvere il problema della ripetizione degli atomi, che si verifica soprattutto con la generazione dei dizionari sulle singole classi. Una possibile soluzione potrebbe essere imporre un vincolo di ortogonalità alla tecnica di dictionary learning utilizzata. Nel complesso, ci sono innumerevoli direzioni in cui cercare di sviluppare la tecnica proposta. La presente dissertazione va quindi considerata come un lavoro esplorativo in un campo particolarmente intricato ed in via di sviluppo.

Bibliografia

- [Alipanahi et al., 2015] Alipanahi, B., Delong, A., Weirauch, M. T., and Frey, B. J. (2015). Predicting the sequence specificities of dna- and rna-binding proteins by deep learning. *Nature Biotechnology*, 33:831–838.
- [Andrew, 1997] Andrew, A. M. (1997). High-level vision: Object recognition and visual cognition, by shimon ullman, mit press (bradford), cambridge, mass., 1996, xviii+412 pp, isbn 0-262-21013-4, (hbk: £33.95). *Robotica*, 15(2):233–236.
- [Andrews et al., 1995] Andrews, R., Diederich, J., and Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Know.-Based Syst.*, 8(6):373–389.
- [Basso et al., 2010] Basso, C., Santoro, M., Verri, A., and Villa, S. (2010). PADDLE: proximal algorithm for dual dictionaries learning. *CoRR*, abs/1011.3728.
- [Biederman, 1987] Biederman, I. (1987). Recognition-by-components: a theory of human image understanding. *Psychological review*, 94 2:115–47.
- [Bishop, 1995] Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA.
- [Bojarski et al., 2017] Bojarski, M., Yeres, P., Choromanska, A., Choromanski, K., Firner, B., Jackel, L. D., and Muller, U. (2017). Explaining how a deep neural network trained with end-to-end learning steers a car. *CoRR*, abs/1704.07911.

- [Chung et al., 2016] Chung, J. S., Senior, A. W., Vinyals, O., and Zisserman, A. (2016). Lip reading sentences in the wild. *CoRR*, abs/1611.05358.
- [Combettes and Pesquet, 2011] Combettes, P. L. and Pesquet, J.-C. (2011). *Proximal Splitting Methods in Signal Processing*, pages 185–212. Springer New York, New York, NY.
- [Danks and London, 2017] Danks, D. and London, A. J. (2017). Algorithmic bias in autonomous systems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI’17, pages 4691–4697. AAAI Press.
- [Doran et al., 2017] Doran, D., Schulz, S., and Besold, T. R. (2017). What does explainable AI really mean? A new conceptualization of perspectives. *CoRR*, abs/1710.00794.
- [Doshi-Velez and Kim, 2017] Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv*.
- [Doshi-Velez et al., 2017] Doshi-Velez, F., Kortz, M., Budish, R., Bavitz, C., Gershman, S., O’Brien, D., Schieber, S., Waldo, J., Weinberger, D., and Wood, A. (2017). Accountability of AI under the law: The role of explanation. *CoRR*, abs/1711.01134.
- [Erhan et al., 2009] Erhan, D., Bengio, Y., Courville, A., and Vincent, P. (2009). Visualizing higher-layer features of a deep network.
- [F.R.S., 1901] F.R.S., K. P. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.
- [Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. *Advances in Neural Information Processing Systems*, 3.
- [Goodman and Flaxman, 2017] Goodman, B. and Flaxman, S. (2017). European union regulations on algorithmic decision-making and a ”right to explanation”. *AI Magazine*, Vol 38, No 3.

- [Hammond and Axelrod, 2006] Hammond, R. A. and Axelrod, R. (2006). The evolution of ethnocentrism. *Journal of Conflict Resolution*, 50(6):926–936.
- [Hoyer, 2004] Hoyer, P. O. (2004). Non-negative matrix factorization with sparseness constraints. *J. Mach. Learn. Res.*, 5:1457–1469.
- [Johnson, 2006] Johnson, S. (2006). *Stephen Johnson on Digital Photography*. O'Reilly Series. O'Reilly Media, Incorporated.
- [Kindermans et al., 2017a] Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., Erhan, D., and Kim, B. (2017a). The (un)reliability of saliency methods. *CoRR*, abs/1711.00867.
- [Kindermans et al., 2017b] Kindermans, P.-J., Schütt, K., Alber, M., Müller, K.-R., Erhan, D., Kim, B., and Dähne, S. (2017b). Learning how to explain neural networks: Patternnet and patternattribution.
- [Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- [Laroche et al., 2015] Laroche, C., Kowalski, M., Papadopoulos, H., and Richard, G. (2015). A structured nonnegative matrix factorization for source separation. *2015 23rd European Signal Processing Conference (EUSIPCO)*, pages 2033–2037.
- [Lecun et al., 1998] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- [LeCun and Cortes, 2010] LeCun, Y. and Cortes, C. (2010). MNIST handwritten digit database.
- [LeCun et al., 1999] LeCun, Y., Haffner, P., Bottou, L., and Bengio, Y. (1999). *Object Recognition with Gradient-Based Learning*, pages 319–345. Springer Berlin Heidelberg, Berlin, Heidelberg.

- [Lee and Sebastian Seung, 1999] Lee, D. and Sebastian Seung, H. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–91.
- [Lee and Seung, 2000] Lee, D. D. and Seung, H. S. (2000). Algorithms for non-negative matrix factorization. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS’00, pages 535–541, Cambridge, MA, USA. MIT Press.
- [Lee et al., 2017] Lee, H., Tajmir, S., Lee, J., Zissen, M., Yeshiwash, B. A., Alkabsab, T. K., Choy, G., and Do, S. (2017). Fully automated deep learning system for bone age assessment. *Journal of Digital Imaging*, 30(4):427–441.
- [Lipton, 2016] Lipton, Z. C. (2016). The mythos of model interpretability. *CoRR*, abs/1606.03490.
- [Logothetis and Sheinberg, 1996] Logothetis, N. K. and Sheinberg, D. L. (1996). Visual object recognition. *Annual Review of Neuroscience*, 19(1):577–621. PMID: 8833455.
- [Lombrozo, 2006] Lombrozo, T. (2006). The structure and function of explanations. *Trends in Cognitive Sciences*, 10(10):464–470.
- [Mohseni and Ragan, 2018] Mohseni, S. and Ragan, E. D. (2018). A human-grounded evaluation benchmark for local explanations of machine learning. *CoRR*, abs/1801.05075.
- [Montavon et al., 2015] Montavon, G., Bach, S., Binder, A., Samek, W., and Müller, K. (2015). Explaining nonlinear classification decisions with deep taylor decomposition. *CoRR*, abs/1512.02479.
- [Montavon et al., 2017] Montavon, G., Samek, W., and Müller, K. (2017). Methods for interpreting and understanding deep neural networks. *CoRR*, abs/1706.07979.

[Nguyen et al., 2016a] Nguyen, A., Yosinski, J., Bengio, Y., Dosovitskiy, A., and Clune, J. (2016a). Plug & play generative networks: Conditional iterative generation of images in latent space. *CoRR*, abs/1612.00005.

[Nguyen et al., 2016b] Nguyen, A. M., Yosinski, J., and Clune, J. (2016b). Multi-faceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *CoRR*, abs/1602.03616.

[Olah et al., 2017] Olah, C., Mordvintsev, A., and Schubert, L. (2017). Feature visualization. *Distill*. <https://distill.pub/2017/feature-visualization>.

[Olshausen and Field, 1997] Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311 – 3325.

[Palmer, 1977] Palmer, S. E. (1977). Hierarchical structure in perceptual representation. *Cognitive Psychology*, 9(4):441 – 474.

[Pasquale, 2015] Pasquale, F. (2015). *The Black Box Society*. Harvard University Press.

[Pedreschi et al., 2018] Pedreschi, D., Giannotti, F., Guidotti, R., Monreale, A., Pappalardo, L., Ruggieri, S., and Turini, F. (2018). Open the black box data-driven explanation of black box decision systems. *CoRR*, abs/1806.09936.

[Ras et al., 2018] Ras, G., van Gerven, M., and Haselager, P. (2018). Explanation methods in deep learning: Users, values, concerns and challenges. *CoRR*, abs/1803.07517.

[Ribeiro et al., 2016] Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). ”why should I trust you?”: Explaining the predictions of any classifier. *CoRR*, abs/1602.04938.

[Silver et al., 2016] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever,

I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

[Simonyan et al., 2013] Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034.

[Szegedy et al., 2013] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. (2013). Intriguing properties of neural networks. *CoRR*, abs/1312.6199.

[Tessitore and Prevete, 2011] Tessitore, G. and Prevete, R. (2011). Designing structured sparse dictionaries for sparse representation modeling. *Burduk R., Kurzyński M., Woźniak M., Żołnierk A. (eds) Computer Recognition Systems 4. Advances in Intelligent and Soft Computing, vol 95. Springer, Berlin, Heidelberg.*

[Wachsmuth et al., 1994] Wachsmuth, E., Oram, M. W., and Perrett, D. I. (1994). Recognition of objects and their component parts: Responses of single units in the temporal cortex of the macaque. *Cerebral Cortex*, 4(5):509–522.

[Wasserman, 2010] Wasserman, L. (2010). *All of Statistics: A Concise Course in Statistical Inference*. Springer Publishing Company, Incorporated.

[Weller, 2017] Weller, A. (2017). Challenges for transparency. *CoRR*, abs/1708.01870.

[Wilson and Keil, 1998] Wilson, R. A. and Keil, F. (1998). The shadows and shallows of explanation. *Minds and Machines*, 8(1):137–159.

[Xiao et al., 2017] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.