



VNIVERSITAT
DE VALÈNCIA

Curs 2023/ 2024

Anàlisi de soroll real i simulat de detectors
d'Ones Gravitacionals mitjançant tècniques de
Machine Learning.

Autor:
Guillem Fernández Rodríguez

Tutora: ISABEL CORDERO CARRIÓN

Índex

1. Introducció	5
2. Ones gravitacionals	7
2.1. Model físic	7
2.2. Funcionament dels detectors	9
3. Tècniques d'anàlisi de senyals	13
3.1. Anàlisi de freqüències	14
3.1.1. Senyals continus	14
3.1.2. Senyals discrets	16
3.2. Processos estocàstics	18
3.3. Blanqueig de soroll	21
3.4. Estimació del <i>PSD</i>	22
3.5. Q-transformació	23
3.6. Espectrogrames	24
4. Machine learning	27
4.1. Fonaments	27
4.1.1. Multicapa de perceptrons	29
4.1.2. Regularització	31
4.1.3. Avaluació i búsqueta d'hiperparàmetres	32
4.2. Xarxes neuronals convolucionals	32
5. Materials i Mètodes	41
5.1. Conjunts de dades	41
5.1.1. Espectrogrames de glitches sintètics	41
5.1.2. Gravity Spy	43
5.2. Mètrica d'avaluació	45
5.3. Desequilibri de classes	46
5.4. Funció de cost modificada	47
5.5. Augmentació d'imatges	47
5.6. Models tipus <i>VGG</i>	48
5.7. Fusió de múltiples punts de vista	48
5.8. Mecanisme d'atenció	52
5.9. Transferència d'aprenentatge	53
6. Resultats	55

6.1. Classificació de <i>glitches sintètics</i>	55
6.2. Classificació de Gravity Spy	57
7. Conclusions	63
Bibliografia	65

Capítol 1

Introducció

La teoria de la relativitat general va revolucionar la concepció de l'espai i el temps a l'univers [1], considerant una geometria que combina l'espai-temps per explicar la gravetat i en què la velocitat de la llum és una constant universal. En particular, segons aquesta teoria, els canvis en la gravetat es propagarien com a ones viatjant a la velocitat de la llum. Aquest fenomen, que anomenem ones gravitacionals, deuria poder mesurar-se com a una distorsió a l'espai-temps. S'hipotetitzava que esdeveniments astrofísics com una supernova o la unió de dos forats negres o púlsars produiria efectes sobre l'espai-temps que arribarien fins a la Terra sense cap interferència.

L'efecte d'aquestes ones però, és tan menut, que es va tardar quasi un segle en desenvolupar detectors amb la sensibilitat suficient. A l'11 de setembre del 2016 es va detectar per primera volta una ona gravitacional, resultat de la unió de dos forats negres. Aquests tenien $36M$ i $29M$ (on M fa referència a masses solars) i la seu unió $62M$ [24], emetent $3M$ en energia a través de radiació gravitatòria en les ones gravitacionals.

La recent capacitat de detectar ones gravitacionals i cada volta a una major distància resulta molt interessant per conèixer millor l'univers i refinjar les teories físiques actuals. El 17 d'agost del 2017 es va detectar l'ona gravitacional corresponent a la unió de dos púlsars, un ànalisi simultani de l'espectre electromagnètic d'aquest esdeveniment va concloure la formació d'elements pesats com a resultat i es va confirmar així l'origen de l'or, platí i urani a l'univers. Els actuals detectors i la següent generació que ampliarà el rang de detecció en un factor de 1000, permetran estudiar la distribució de diferents objectes astrofísics com sistemes binaris de forats negres i les seus característiques. Futurs detectors captaran potser l'efecte cosmològic de fons del Big Bang.

Les raons i possibilitats que ofereix aquest camp d'estudi són immenses, i el tipus d'obstacles als quals s'enfronta són diversos. Cal construir detectors d'altíssima precisió, aplicar diferents estratègies d'ànalisi de dades i guiar aquests processos amb modelització astrofísica. En aquest treball ens centrarem en un dels engranatges de la maquinària utilitzada per detectar ones gravitacionals, i un enfocament dels esforços actuals per millorar la qualitat de les dades detectades.

Les dades obtingudes als detectors principals contenen gran quantitat de soroll, complexe en el seu comportament en el temps i contingut de freqüències. Aquest és a sovint tractat de forma ideal com a un soroll estacionari i Gaussià, però sempre tenint en compte les possibles desviacions respecte d'aquest comportament. Hi ha, efectivament, gran quantitat de soroll de comportaments no estacionaris de diversos orígens (a voltes els propis sistemes de detecció, sorolls ambientals o d'origen desconegut), un tipus de soroll transitori de curta duració que anomenem *glitches* que poden ser confosos amb ones gravitacionals.

Els *glitches* poden ser caracteritzats i detectats per models de *machine learning*, però per entrenar aquest tipus de models cal disposar d'una gran quantitat de dades correctament classificades. Per aconseguir aquests conjunts de dades s'han realitzat projectes de col·laboració ciutadana com Gravity Spy i GwitchHunters, en què

voluntaris poden aprendre a identificar *glitches* i classificar dades que possiblement en contenen. Grans esforços per part dels membres de les diverses col·laboracions (LIGO, Virgo, KAGRA) han realitzat l'organització e implementació efectiva d'aquests projects, els quals han donat a lloc conjunts de dades massius i de molta qualitat.

Els *glitches* poden ser resultat de diversos factors associats als instruments del detector i el mateix ambient en què es troben. Així, és útil utilitzar dades de canals auxiliars i estudiar la seua correlació amb el canal principal. Actualment, el projecte GwitchHunter està construint un conjunt de dades de *glitches* classificats en què s'inclouen diversos canals auxiliars. En aquest treball, explorarem l'aplicació de diversos models de machine learning en preparació a l'estudi d'aquestes dades i ens centrarem en el canal principal del qual hi ha moltes dades accessibles i de bona qualitat.

Explicarem, en primer lloc, el model físic d'ones gravitacionals, les diverses tècniques d'anàlisi de senyals que aplicarem i una introducció al deep learning i al tipus de models que gastarem: les xarxes neuronals convolucionals. Realitzarem experiments tant sobre dades simulades com sobre un conjunt de dades reals del projecte Gravity Spy, explorant els diversos models i tècniques utilitzades en la literatura científica recent. Incloureem tot el codi utilitzat per generar les figures d'aquest document així com per entrenar els models en un repositori¹ disponible per al lector.

¹<https://github.com/introspective-swallow/Gravitational-Waves-Glitch-Classification>

Capítol 2

Ones gravitacionals

2.1. Model físic

La teoria de la relativitat general explica les ones gravitacionals a conseqüència d'acceleracions de masses o energia, com a un canvi al moment quadrupol o moments superiors de la mètrica de l'espai temps ([47] Ch. 5). Com que les ones gravitacionals que podem detectar provenen d'events astrofísics a milions o milers de milions d'anys llum de la terra, tenen un efecte molt feble en les nostres mesures i podem modelar-les com a una perturbació sobre la geometria de l'espai temps.

A continuació expliquem superficialment com es modelen les ones gravitacionals amb el formalisme de la relativitat general, basant-nos en ([17] Ch 9.1). El model més simple per descriure una perturbació xicoteta sobre l'espai temps és la gravetat linearitzada o aproximació per a camps de gravitació débil.

En aquesta secció utilitzarem el conveni de sumació de Einstein. Considerem una mètrica de fons coneguda g_{ab}^B , com per exemple la mètrica plana de Minkowski $g_{ab}^B = \text{diag}(-1, 1, 1, 1)$, aleshores podem modelar la seua perturbació h_{ab} amb la condició

$$g_{ab} = g_{ab}^B + h_{ab}, \quad |h_{ab}| \ll 1$$

que resulta en una aproximació molt precisa si el sistema de referència es troba a gran distància de l'origen de les perturbacions, com és el cas dels observatoris en la terra. Ens podem fixar que la condició sobre h_{ab} descarta els termes d'ordre superior al lineal.

Ara es pot definir el que s'anomena la perturbació amb traça inversa

$$\bar{h}_{ab} = h_{ab} - \frac{1}{2}g_{ab}^B h_c^c,$$

utilitzar unes coordenades de forma que es complisca la condició de *Lorentz Gauge*

$$\nabla_a \bar{h}^{ab} = 0 \tag{2.1}$$

i en aquest cas les equacions de Einstein en el buit es redueixen a una equació de ona

$$\nabla^c \nabla_c \bar{h}_{ab} = 0.$$

A més a més aquesta condició no determina a \bar{h}_{ab} de forma única, ja que es poden aplicar transformacions de *gauge* infinitesimals que mantenen aquesta condició. Si incloem la condició de *transverse-traceless gauge*,

$$\bar{h}_{a0}^{TT} = 0 \quad \text{i} \quad h_a^{TTa} = 0.$$

Ens podem fixar que \bar{h}_{aa}^{TT} implica que la pertorbació és únicament espacial i $\bar{h}_a^{TTa} = 0$ que $h_c^c = 0$ de forma que $\bar{h} = h$ són intercanviables. Amb aquesta condició i (2.1), imosem vuit restriccions sobre els deu graus de llibertat de h_{ab} . Els dos graus de llibertat restants es corresponen a dos possibles polaritzacions de les ones gravitacionals. Així podem descomposar h_{jk}^{TT} en dos tensors e_{ij}^+, e_{ij}^\times i amplituds h_+, h_\times

$$h_{jk}^{TT} = h_+ e_{ij}^+ + h_\times e_{ij}^\times.$$

Si la ona gravitacional es propaga en la direcció de l'eix z , tindriem

$$h_{jk}^{TT} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & h_+ & h_\times & 0 \\ 0 & h_\times & h_+ & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

A partir d'aquestes equacions, es pot deduir que el seu efecte sobre dues partícules en caiguda lliure a distància L és un canvi en la distància ΔL de forma que $h \sim \frac{\Delta L}{L}$ on h és l'amplitud de l'ona gravitacional. Per fer-se una idea de l'ordre d'aquesta magnitud, un esdeveniment que emet $1M$, durant $1ms$ a $15Mpc$ produirà una deformació $h \sim 5 \times 10^{-22}$ ([47] Ch. 1.4). Aquest valor ha sigut de fet un important umbral de sensibilitat que els detectors moderns han aconseguit superar i a partir del qual es van començar a detectar els primers esdeveniments gravitacionals.

L'esdeveniment que més a sovint es detectat és el generat per un sistema binari de forats negres. És especialment simple de modelar ja que es tracta d'un espai-temps amb curvatura extrema però sense contingut al tensor d'energia-moment, de forma que els graus de llibertat del sistema són molt menors a altres com els sistemes binaris de púlsars. Existeixen diverses famílies d'ones per modelar aquests events, una ona gravitacional generada per un sistema binari de forats negres pot ser modelada a partir de 15 paràmetres [23].

- Intrínsecos: 2 de les masses dels objectes en òrbita, 6 paràmetres dels dos spins dels objectes (magnitud i orientació)
- Extrínsecos: la distància des de l'origen al detector, 2 coordinades amb la posició al cel, l'angle d'inclinació, l'angle de polarització, el temps de referència i la fase orbital en aquest.

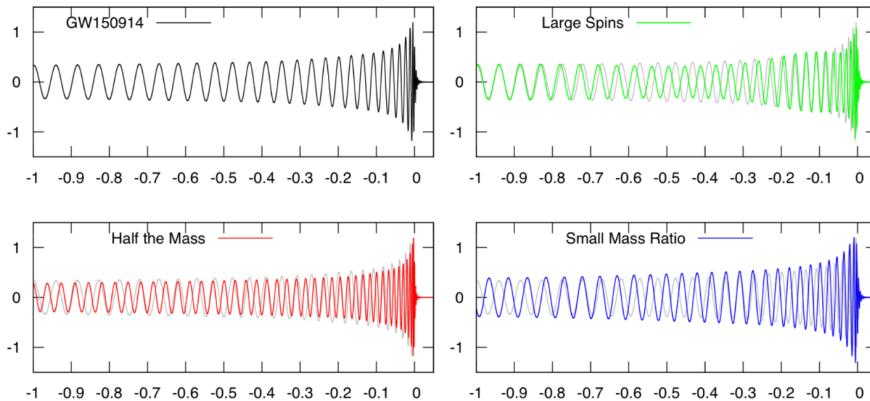


Figura 2.1: Exemple de l'efecte de variar diversos paràmetres sobre l'ona gravitacional modelada per la detecció de l'esdeveniment GW150914. Imatge produïda per N. Cornish i T. Littenberg extreta de [26].

Podem mesurar el seu efecte sobre un grup de partícules posicionades en un pla perpendicular a l'ona com a una oscil·lació en forma de creu amb les dues possibles polaritzacions h_+, h_\times (vegeu Figura 2.2).

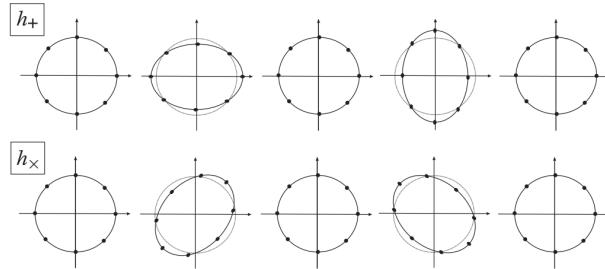


Figura 2.2: Efecte del pas d'una ona gravitacional transversal a un grup de partícules distribuïdes en un cercle. La primera fila mostra l'efecte d'una ona amb polarització positiva i la segona amb polarització en creu, exagerant la magnitud de deformació per facilitar la seu visualització. Imatge extreta de ([47] pag. 55).

2.2. Funcionament dels detectors

Per mesurar l'efecte de les ones gravitacionals, cal mesurar la deformació a l'espai. Per fer-ho la tècnica més efectiva i que permet fer mesures amb suficient precisió és la interferometria làser, en què es mesura la distància que recorre la llum per diversos camins òptics al llarg del temps. Per fer-ho, s'emet un làser en dues direccions diferents, que eventualment és reflectida i en trobar-se al punt d'eixida produueix un patró d'interferència. Aquest patró depén de la distància recorreguda en cadascun dels dos camins diferents, de forma que un canvi en aquesta deguda a una ona gravitacional pot ser detectat.

Els detectors LIGO i Virgo són interferòmetres de Michelson, en què un làser produueix un feix de llum que és dividit en dos camins o braços ortogonals amb un divisor de feix. Al final dels camins, dos espills reflecteixen els làsers que es tornen a trobar al divisor de feix. Els braços estan ajustats de forma que l'interferència dels dos feixos de llum siga destructiva en la direcció del fotodiode. Quan les longituds són diferents l'interferència no es totalment destructiva i part de la llum travessera cap al fotodíode, generant un patró d'interferència (2.3).

Suposem que tenim dos braços de longituds l_1, l_2 , aleshores el patró d'interferència dependrà de $l_1 - l_2$ i el canvi en aquest patró de $\Delta l = \Delta l_1 - \Delta l_2$. Així, podem mesurar la deformació

$$h = \frac{\Delta l}{l}$$

amb l la longitud dels braços sense cap deformació. Podem observar un canvi al patró d'interferència amb un canvi de l'ordre de la longitud d'ona del làser $\Delta l \sim \lambda_{\text{laser}}$. Considerem els braços del detector de Virgo amb $l = 3\text{km}$ i un làser infraroig $\lambda_{\text{laser}} = 1\mu\text{m}$, així tenim una sensitivitat

$$h = \sim \frac{\lambda_{\text{laser}}}{l} = \frac{10^{-6}}{10^3} = 10^{-9},$$

que es troba molt lluny de l'ordre desitjat $h \sim 10^{-20}$.

Diverses tècniques són utilitzades per millorar aquesta sensitivitat als detectors actuals. Per exemple podem ampliar el recorregut de la llum en cada braç fent-la rebotar amb una cavitat de Fabry-Pérot i aconseguir una

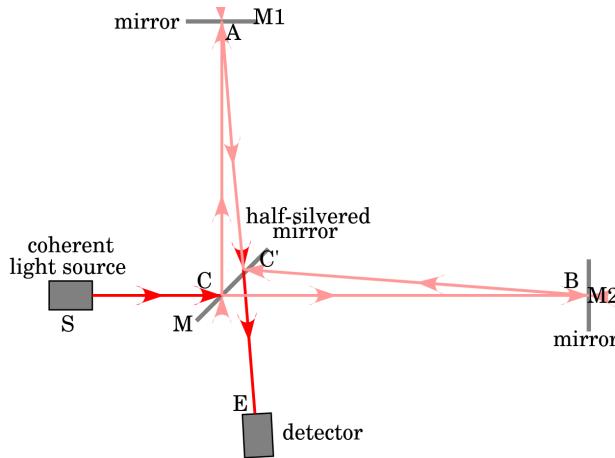


Figura 2.3: Representació del camí òptic d'un interferòmetre de Michelson. Imatge extreta de [Wikipedia](#).

longitud efectiva major l_{eff} . Ara, per tal que la llum recórrega el braç amb la mateixa deformació en entrar i eixir, cal que la seu longitud efectiva no siga de l'ordre de la longitud d'ona de les ones gravitacionals esperades. Si aquestes tenen freqüència f_{GW} , la longitud d'ona és $\lambda_{GW} = \frac{c}{f_{GW}}$. Si busquem $f_{GW} = 300Hz$, $\lambda_{GW} = 1000km$ que equival a una longitud 300 voltes major a cada braç, i aconseguim una sensitivitat de l'umbral de detecció de

$$h \sim 10^{-12}.$$

Una altra tècnica és millorar la sensitivitat en detectar canvis al patró d'interferència. Si captem aquest patró amb un fotodiode, esperem que detecte N_f fotons en un període $\tau \sim \frac{1}{f_{GW}}$ de temps de l'ordre del període d'una longitud d'ona. Ara podem considerar N_f com a una variable aleatòria que segueix un procés de Poisson, com al nombre de partícules que arriben de forma independent en el període de temps τ . Així, la desviació típica o fluctuació d'aquesta variable és de $\sqrt{N_f}$. D'aquesta forma, un canvi $\Delta l \sim \frac{\sqrt{N_f}}{N_f} \lambda_{laser}$ de l'ordre de la proporció de la fluctuació natural per longitud d'ona implica que una distribució diferent induïda per un canvi de les longituds dels braços. Si un làser té potència P_{laser} (energia transmessa per temps) i un fotó té energia $\frac{hc}{\lambda_{laser}}$ obtenim $N_f = \frac{P_f}{hc/\lambda_{laser}} \tau$. Així un làser infraroig amb $P_{laser} = 1W$ i les mateixes condicions anteriors ens donen

$$h \sim \frac{N_f^{-\frac{1}{2}} \lambda_{laser}}{l} = 10^{-20}$$

Els interferòmetres avançats actuals són interferòmetres de Michelson a gran escala amb sistema de reciclatge dual (2.4). Aquest nom ve de l'ús de dues tècniques en què es capture el làser en diverses eixides i torna a ser introduït al detector.

Prodir un làser d'alta potència permet millorar la sensitivitat, però dona altres problemes com soroll del calfament dels espills, deformacions en la geometria dels espills o una major quantitat de llum difusa que pot produir interferències. Enlloc d'augmentar la potència del laser excessivament, s'utilitza un sistema de reciclatge de potència per aconseguir el mateix. Si col·loquem un espill en l'eixida simètrica per tornar a introduir la llum que eixiria a l'interferòmetre, podem augmentar la potència del làser, que normalment resulta en un ordre de magnitud de sensitivitat.

També podem usar reciclatge de senyals, incloent-hi un espill en l'eixida antisimètrica. D'aquesta forma se sol poder augmentar la sensibilitat en certes bandes de freqüència.

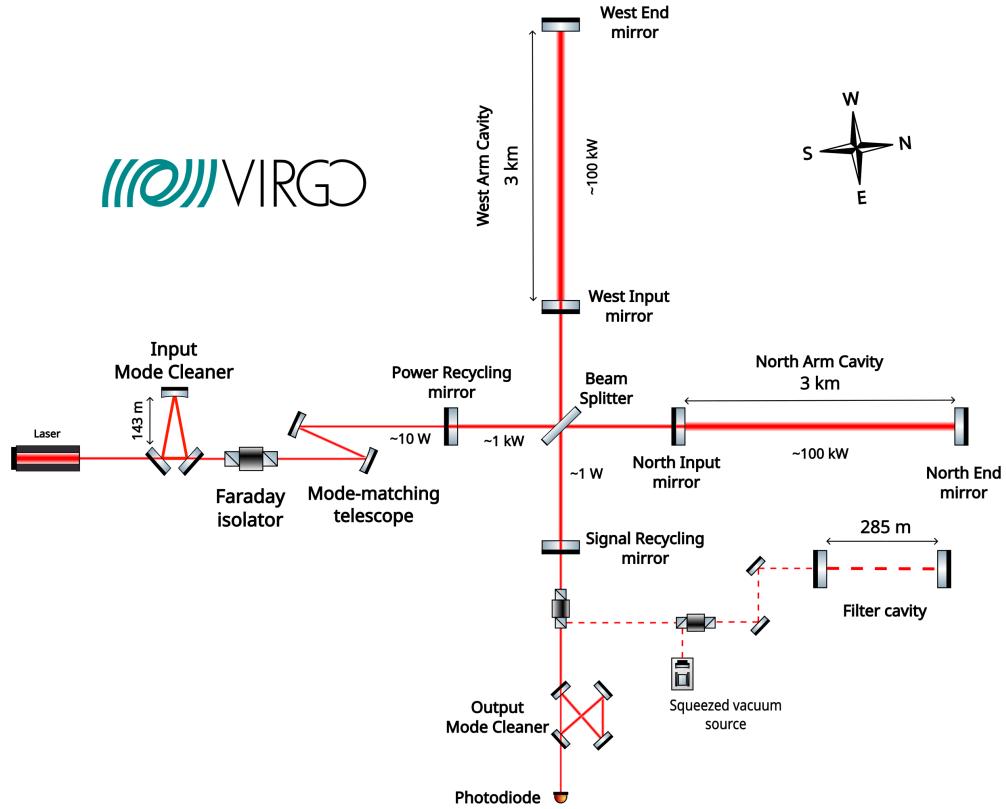


Figura 2.4: Representació dels components i camí òptic del interferòmetre de Virgo durant el període de detecció *O4* (2023-2024). Imatge extreta de [Wikipedia](#).

Addicionalment, al detector avançat de Virgo gasta dos sistemes addicionals (2.4). Té un sistema de neteja d'entrada (*Input Mode Cleaner*), per millorar la qualitat del làser d'entrada i un sistema de reducció de soroll quàntic (*Quantum Noise Reduction*), necessari per sorolls produïts per la naturalesa quàntica de la llum.

Tots aquests sistemes a més a més estan perfeccionats i aïllats per una màxima efectivitat. Els espills estan polits a escala atòmica de forma que només es perd un 0,01 % de llum per cada 6km. Tots els components òptics estan suspesos per evitar moviments del terra i es troben al buit per ignorar soroll acústic i pols. El feix de llum d'eixida és reduït per un factor de 40 per ser compatible amb el fotodiòde i travessa un sistema de neteja final per filtrar dades de mala qualitat (*Output Mode Cleaner*). Tots aquests sistemes permeten aconseguir una sensibilitat suficient per a detectar ones gravitacionals des de 10Hz fins a 10KHz aproximadament i les pròximes modificacions permetran encara millor sensibilitat. En (2.5) podem observar la sensibilitat per freqüència mesurada en els últims períodes de detecció i la projecció de les futures O4 i O5

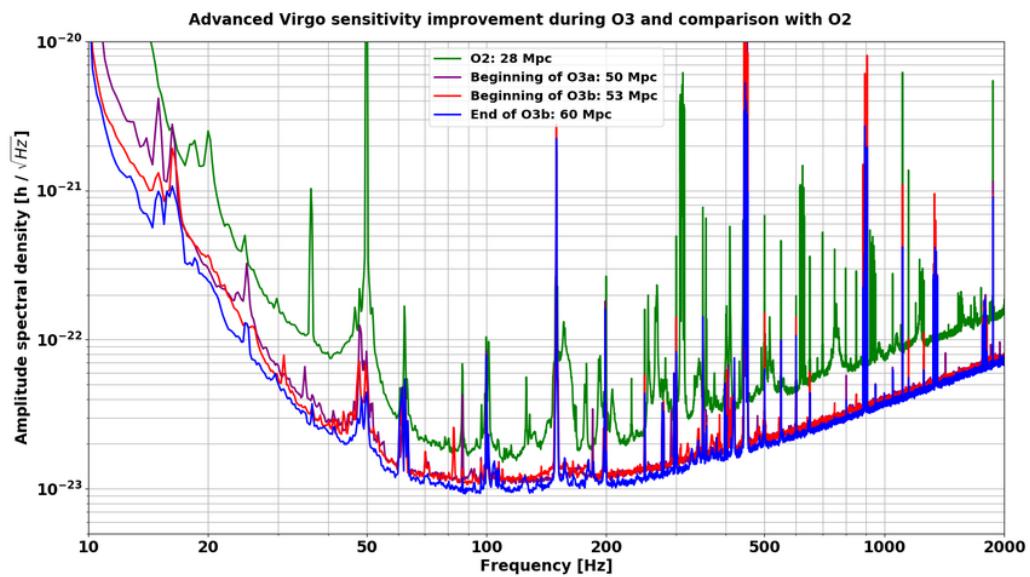


Figura 2.5: Sensibilitat en les diferents freqüències del detector de Virgo durant els diferents períodes de recollida de dades *O2* en verd, principi de *O3a* en morat, principi de *O3b* en roig i final de *O3b* en blau. Imatge extreta de [43].

Capítol 3

Tècniques d'anàlisi de senyals

Per estudiar ones gravitacionals treballem amb la deformació $h(t)$ respecte al temps, i altres mesures de canals auxiliars. Es tracta de variables que varien amb el temps de forma contínua, que en l'àrea del processament de senyals digitals anomenem senyals analògics. La referència principal d'aquesta secció és [9]. Ara a l'hora d'estudiar-les cal mesurar-les en diferents instants en el temps, és a dir cal discretitzar-les i convertir-les en senyals digitals. Així donada un senyal analògica

$$\begin{aligned}x_a : \mathbb{R} &\longrightarrow \mathbb{R} \\t &\mapsto x_a(t)\end{aligned}$$

podem discretitzar-la mesurant valors a intervals de temps equidistants per obtindre una sèrie temporal

$$\begin{aligned}x : \mathbb{Z} &\longrightarrow \mathbb{R} \\n &\mapsto x(n) = x_a(nT) = x_a\left(\frac{n}{F_s}\right)\end{aligned}$$

per a $T > 0$ l'interval de temps entre cada valor o equivalentment $F_s = \frac{1}{T}$ la freqüència de mostreig.

Com a exemple considerem la senyal $x_a(t) = \sin(2\pi t)$ i la seu discretització $x(n) = \sin(2\pi n/F_s)$ triant una freqüència de mostreig de $F_s = 13\text{Hz}$ (3.1).

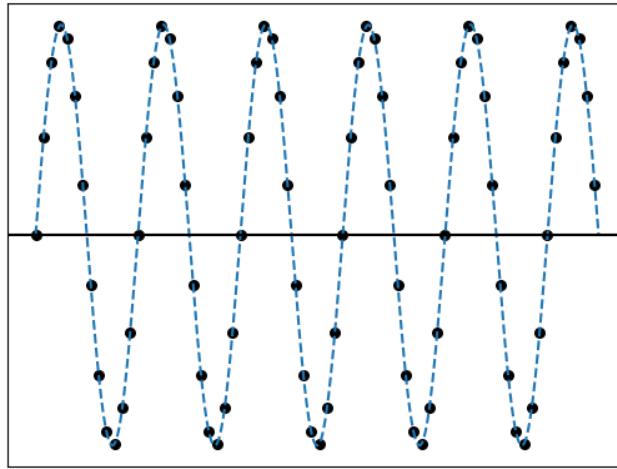


Figura 3.1: Discretització d'un senyal sinoidal de freqüència 1Hz amb freqüència de mostreig 13Hz.

A banda de mostrejar un senyal analògic, per convertir-la en format digital també és necessari quantitzar els valors que pren i codificar cada valor discret amb un valor binari per poder processar-ho digitalment. El procés de quantització origina sempre un error, i cal ser prudent a l'hora de triar una codificació adient per a la precisió requerida. Al llarg d'aquesta secció ignorarem aquests dos processos, ja que l'error és molt reduït amb la precisió gastada per ordinadors actuals i parlarem de senyals digitals o sèries temporals de forma intercanviable.

Cal, en canvi, considerar la pèrdua d'informació causada pel procés de discretització, per a això primer cal estudiar els senyals digitals treballant des del punt de vista de les freqüències que contenen.

3.1. Anàlisi de freqüències

3.1.1. Senyals continuos

L'anàlisi de les freqüències és la descomposició de senyals en funcions sinusoidals o exponencials complexes, una classe de funcions periòdiques. Així, podem passar d'estudiar un senyal en termes d'amplitud respecte al temps a treballar en termes de freqüència, fase i amplitud dels components periòdics en que es descomposa.

Comencem considerant el cas continu. Considerem un senyal analògic (continu) de període $T > 0$, $x(t+T) = x(t), \forall t \in \mathbb{R}$, aleshores aprofitant que és periòdic podem tenir la idea de descomposar-la en les funcions trigonomètriques del mateix període $\sin(2\pi kt/T), \cos(2\pi kt/T)$ per a $k \in \mathbb{N}$, o equivalentment en funcions $e^{i2\pi kt/T}$ per a $k \in \mathbb{Z}$. Per fer-ho definim la sèrie de Fourier.

Definició 3.1.1. Definim la sèrie de Fourier d'un senyal analògic $x(t)$ de període T com

$$\sum_{k=-\infty}^{\infty} c_k e^{i2\pi kt/T}$$

on $c_k = \frac{1}{T} \int_{-T/2}^{T/2} x(t) e^{-i2\pi kt/T} dt$.

Quan treballem amb senyals d'origen físic, en la pràctica aquesta sèrie sempre estarà definida i serà equivalent a l'original, però existeixen diverses condicions que podem imposar sobre la classe de funcions i que resulten raonables si provenen d'un sistema físic.

Teorema 3.1.1. *Siga un senyal analògic $x(t)$ de periode T . Si es compleixen les condicions de Dirichlet:*

1. $x(t)$ és contínua excepte en un nombre finit de punts
2. $x(t)$ té un nombre finit de màxims i mínims
3. $\int_{-T_p}^{T_p} |x(t)| < \infty$ per a tot $T_p > 0$

Aleshores la seua sèrie de Fourier convergeix en $x(t)$, excepte en les seues discontinuïtats en que concergeix en el punt mitjà dels límits laterals.

Doncs quan existeix aquesta descomposició, podem parlar de la freqüència k d'un senyal periòdic referenciant el corresponent coeficient c_k .

Ara, com que normalment tractarem amb senyals aperiòdiques o amb periode desconegut, cal definir un altre tipus de descomposició. Si tenim una senyal no periòdica, podem definir la transformada de Fourier que també existirà baix certes condicions i una transformada inversa per reconstruir el senyal.

Definició 3.1.2. Siga un senyal analògic $x(t)$ que compleix les condicions de Dirichlet (3), aleshores existeix i podem definir la seu transformada de Fourier $X : \mathbb{R} \rightarrow \mathbb{C}$

$$X(F) = \int_{-\infty}^{\infty} x(t)e^{-i2\pi F t} dt$$

i la transformada inversa

$$x(t) = \int_{-\infty}^{\infty} X(F)e^{i2\pi F t} dF$$

Direm que un senyal analògic aperiòdic té una freqüència F si la corresponent transformada de Fourier $X(F) \neq 0$ i per tant que té una freqüència màxima F_{max} si $X(F) = 0$ per a $|F| > F_{max}$ (notem que $X(-F) = \overline{X(F)}$, pel que té sentit aquesta definició).

Podem així estudiar el contingut de freqüències d'una senyal aperiòdic a través de la transformada de Fourier i idealment volem conservar aquesta informació en la discretització del senyal analògic.

Conservarem més o menys informació d'una senyal segons la freqüència màxima d'aquesta que podem arribar a reconstruir a partir de la seuva discretització. El rang de freqüències que podem conservar depén de la freqüència de mostreig que triem, i aquesta relació ve donada per el teorema de Whittaker–Nyquist–Shannon [2].

Teorema 3.1.2 (Nyquist-Shanon). *Donat un senyal analògic x_a amb freqüència màxima $F_{max} = B$, si obtenim una sèrie temporal x amb una freqüència de mostreig $F_s > 2B$ aleshores aquesta determina a x_a .*

Demostració. Podem expressar x_a utilitzant la transformada inversa i com que té una freqüència màxima.

$$x_a(t) = \int_{-\infty}^{\infty} X(F)e^{i2\pi F t} dt = \int_{-B}^{B} X(F)e^{i2\pi F t} dt$$

Si considerem la senyal mesurada amb freqüència de mostreig $F_s = 2B$

$$x(n) = x_a\left(\frac{n}{2B}\right) = \int_{-B}^{B} X(F)e^{i2\pi Fn/2B} dt$$

notem que multiplicant per $\frac{1}{2B}$ obtenim a la dreta el coeficient c_n de la sèrie de Fourier de $X(F)$. Així es clar que $x(n)$ amb $F_s = 2B$ determina a $X(F)$ el qual determina a $x_a(t)$. \square

Anomenem a la freqüència $2B$ la taxa de Nyquist, i en general donat una freqüència de mostreig F_s , només que considerem vàlides les freqüències menors a $F_s/2$. Les freqüències majors que $F_s/2$ poden ser degudes a l'aliàsing, distorsions degudes al mostreig en què una freqüència baixa pot ser confosa amb una alta (3.2). Així és estàndard aplicar tècniques d'antialiàsing per mitigar o eliminar les freqüències majors que $F_s/2$. Per exemple, el detector de Virgo utilitza una freqüència de mostreig de $20kHz$ i el rang vàlid que es va utilitzar durant $O2$ va ser de $10Hz$ a $8kHz$, respectant la taxa de Nyquist.

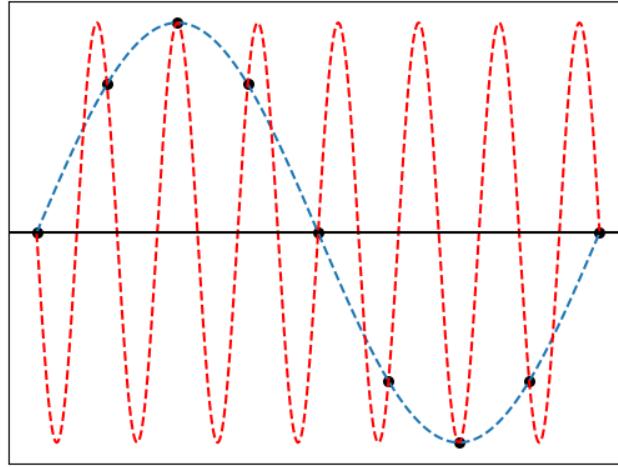


Figura 3.2: Efecte d'aliàsing, considerem un senyal amb freqüència $-7/8$ (corba roja), amb una freqüència de mostreig de $1Hz$. Aleshores recuperem un senyal amb freqüència $1/8$ (corba blava).

3.1.2. Senyals discrets

Si treballem amb un senyal discret amb periode $N \in \mathbb{N}$, aleshores podem expressar-la en el que s'anomena una sèrie de Fourier discreta

Definició 3.1.3. Siga un senyal discret $x(n)$ de periode $N \in \mathbb{N}$, aleshores definim la seua sèrie de Fourier discreta

$$\sum_{k=0}^{N-1} c_k e^{i2\pi kn/N}$$

on $c_k = \frac{1}{N} \sum_{l=0}^{N-1} x(l) e^{-i2\pi l N}$

Notem que en aquest cas com que només que hi ha N punts per període només que hi ha un nombre finit de freqüències que pot contindre. Així només incloem termes fins a $e^{i2\pi k(N-1)N}$ i no cal incloure condicions de convergència ja que podem calcular els coeficients directament:

$$\sum_{n=0}^{N-1} x(n) e^{-i2\pi kn/N} = \sum_{n=0}^{N-1} \left(\sum_{l=0}^{N-1} c_l e^{i2\pi ln/N} \right) e^{-i2\pi kn/N} = \sum_{n=0}^{N-1} \sum_{l=0}^{N-1} c_l e^{i2\pi(l-k)n/N} = N c_k$$

on hem utilitzat que

$$\sum_{l=0}^{N-1} e^{i2\pi ml/N} = \begin{cases} N & \text{si } m = 0 \\ 0 & \text{si no} \end{cases}$$

calculant la suma de la sèrie geomètrica al segon cas. Així obtenim

$$c_k = \frac{1}{N} \sum_{l=0}^{N-1} x(l) e^{-i2\pi ln/N}.$$

Per a un senyal no periòdic, podem definir la transformada de Fourier discreta en el temps o *DTFT*.

Definició 3.1.4. Siga un senyal digital $x(n)$ aperiòdic, definim la seu *DTFT* com

$$X(w) = \sum_{n=-\infty}^{+\infty} x(n) e^{iwn}$$

per a $w \in \mathbb{R}$.

Donem també condicions per la convergència d'aquesta sèrie.

Teorema 3.1.3. La *DTFT* d'un senyal $x(n)$ convergeix a $x(n)$ per a cada $n \in \mathbb{Z}$ si es compleix la condició

$$\sum_{n=-\infty}^{+\infty} |x(n)| < \infty$$

I en aquest cas podem definir la transformada inversa

$$x(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(w) e^{iwn} dw$$

Quan tenim una senyal aperiòdica de duració finita, és a dir zero excepte en un rang de duració finita L , ($x(n) = 0, n < 0, n > L$), aleshores utilitzem la transformada de Fourier discreta o *DFT*.

Definició 3.1.5. Siga un senyal $x(n)$ aperiòdic de duració finita per a $n = 0, 1, \dots, L-1$, definim la seu *DFT* d'ordre N com

$$X(k) = \frac{1}{N} \sum_{n=0}^{L-1} x(n) e^{-i2\pi kn/N}, \quad k = 0, 1, \dots, N-1$$

i la corresponent transformada inversa o *iDFT* com

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{i2\pi kn/N}, \quad n = 0, \dots, L-1$$

Si $N > L$, podem interpretar la *DFT* com a calcular la *DTFT* als valors $w = -2\pi k/N, k = 0, 1, \dots, N-1$ i reconstruir la senyal en el rang $n = 0, 1, \dots, L-1$ gastant aquests.

També podem veure que el procés és equivalent obtindre la sèrie de Fourier discreta del senyal $x_p(n)$ de periode N , que en cada periode pren els valors

$$x_p(n) = \begin{cases} x(n), & 0 \leq n \leq L-1 \\ 0, & L \leq n \leq N-1 \end{cases}$$

i assignant $X(k) = N c_k$.

En la pràctica, si analitzem una serie temporal $x(n)$, disposem només d'una mostra de duració finita L d'aquesta. Per analitzar-la, podem considerar la sèrie temporal

$$\hat{x}(n) = x(n)w(n)$$

on

$$w(n) = \begin{cases} 1, & 0 \leq n \leq L - 1 \\ 0, & \text{en altre cas} \end{cases}$$

i calcular la DFT per a $N \geq L$. Aquest tipus de funció que es gasta per multiplicar una senyal s'anomena funció finestra en el camp del processament de senyals (aquesta es tracta d'una finestra rectangular), es caracteritza per ser zero excepte en un interval d'interés. L'ús de funcions finestra però, dona lloc a un fenòmen anomenat fuga espectral (*spectral leakage*) per el qual la potència de les freqüències originals es distribuïda en un rang de freqüències en la sèrie \hat{x} . Per reduir aquest efecte altres funcions de finestra $w(n)$ poden ser utilitzades (a sovint a canvi d'una menor resolució de freqüències), com la funció de finestra Hann o de la Tukey (3.3).

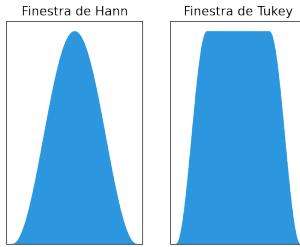


Figura 3.3: Exemples de funcions finestra.

Com a exemple mostrem el DFT amb una finestra rectangular i la finestra de Hann (3.4) aplicades sobre el senyal $x(n) = \cos(\frac{\pi}{2}n)$ amb $L = 25$ i $N = 512$.

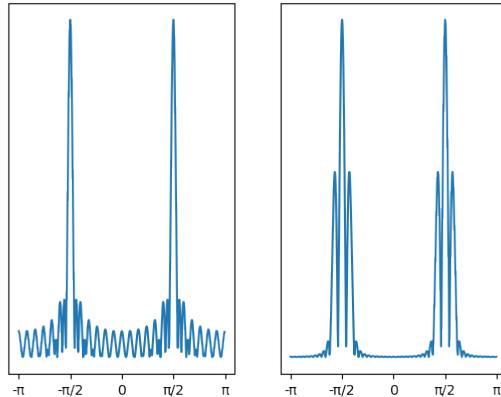


Figura 3.4: DFT d'un senyal discret en que s'ha aplicat una funció de finestra rectangular a l'esquerra i una finestra Hann a la dreta.

3.2. Processos estocàstics

Per modelar el soroll dels detectors d'ones gravitacionals considerem les senyals analògiques com a processos estocàstics. Formalitzem primer els conceptes principals del processos estocàstics adaptant definicions i resultats

del llibre [6] i desenvolupem després les propietats que ens resulten interessants de forma similar a com es fa en ([9] Ch. 12).

Definició 3.2.1. Un procés estocàstic és una família de variables aleatòries $\{X_t, t \in T\}$ definides en un espai de probabilitat (Ω, \mathcal{F}, P) , per a un conjunt d'índex $T \subseteq \mathbb{Z}$.

Com que treballem amb senyals discrets amb valors reals, suposarem que $\Omega = \mathbb{R}$ i $T = \mathbb{Z}$. Denotem així un procés estocàstic per $\{X_n, n \in \mathbb{Z}\}$ i un mostra d'aquest per $\{x_n, n \in A\}$, per a $A \subseteq \mathbb{Z}$. Anomenarem serie temporal tant al procés estocàstic com a una mostra d'aquest.

Així, representem la combinació de tots els sorolls detectats en N mostres de temps com a un procés estocàstic $\{X_t, t \in T\}$ amb distribució de probabilitat conjunta $P(X)$. La mitjana d'una variable aleatoria es defineix per

$$E[X_n] = \int_{-\infty}^{\infty} x_n P(x_n) dx_n$$

Podem definir la mitjana de cada variable $E[X_n]$, la funció d'autocorrelació i la funció d'autocovariància per analitzar el comportament d'aquests processos.

Definició 3.2.2. Siga $\{X_t, t \in T\}$ un procés de forma que $Var(X_t) < \infty$ per a cada $t \in T$, aleshores definim la funció d'autocorrelació $\gamma_{xx} : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{R}$ per

$$\gamma_{xx}(i, j) = E[X_i X_j]$$

per a $i, j \in T$.

Definim la funció d'autocovariància $c_{xx} : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{R}$

$$c_{xx}(i, j) = Cov(X_i, X_j) = E[(X_i - E[X_i])(X_j - E[X_j])]$$

De forma que és fàcil veure que $c_{xx}(i, j) = \gamma_{xx}(i, j) - E[X_i]E[X_j]$ i per tant coincideixen quan les variables tenen mitjana zero.

En la pràctica només que disposarem d'una mostra del procés estocàstic doncs cal garantir que aquesta és suficient per estimar els valors d'interès. Per fer-ho cal assumir més propietats sobre els processos que estudiarem.

Definició 3.2.3. Una sèrie temporal $\{X_t, t \in \mathbb{Z}\}$ és estacionària si

1. $E[|X_t|^2] < \infty$ per a tot $t \in \mathbb{Z}$
2. $E[X_t] = m_x$ per a tot $t \in \mathbb{Z}$
3. $c_{xx}(i, j) = c_{xx}(i + h, j + h)$ per a tot $i, j, h \in \mathbb{Z}$

Així en una sèrie temporal estacionària la mitjana és constant en el temps, té un segon moment ben definit per cada variable i la funció d'autocovariància només depén de la diferència temporal $\tau = i - j$. Podem així redifinir les funcions anteriors aplicant aquesta propietat.

$$c_{xx}(i, j) = c_{xx}(i - j, 0) = c_{xx}(\tau)$$

i

$$\gamma_{xx}(i, j) = c_{xx}(i, j) + m_x^2 = c_{xx}(\tau) + m_x^2 = \gamma_{xx}(\tau)$$

Notem que aquestes funcions són parelles per la simetria respecte als índex, $\gamma_{xx}(\tau) = \gamma_{xx}(i, j) = \gamma_{xx}(j, i) = \gamma_{xx}(-\tau)$ i anàlogament per a c_{xx} .

Ara estem en condicions d'intentar estimar aquestes propietats amb una mostra.

Definició 3.2.4. Donada una mostra $\{x_n, n \in \mathbb{Z}\}$, definim la mitjana temporal per a un $N \in \mathbb{Z}$ com

$$\hat{X}^N = \frac{1}{2N+1} \sum_{n=-N}^N x(n)$$

i la estimació de l'autocorrelació

$$r_{xx}^N(m) = \frac{1}{2N+1} \sum_{n=-N}^N x(n)x(n+m)$$

per a $m \in \mathbb{Z}$.

Per veure que aquests estimadors són bons, volem que s'aproximen als valors reals quan N tendeix cap a infinit. La forma en que mesurem aquesta propietat és la convergència en mitjana quadrada, que implica convergència en probabilitat.

Definició 3.2.5. Siga \hat{X}_N un estimador de la variable aleatòria X . Direm que \hat{X}^N convergeix en mitjana quadrada a X si

$$\lim_{N \rightarrow +\infty} E[|\hat{X}^N - X|^2] = 0$$

En particular ens interessa que l'estimació de la mitjana i de l'autocorrelació tinga aquest comportament, i en aquest cas direm que el procés és ergòdic respecte a aquests valors.

Una condició suficient és que el l'estimador siga no esbiaixat $E[\hat{X}_N] = E[X]$ i que la seu variància convergisca a zero.

Podem veure la primera propietat fàcilment en ambdós casos.

$$\begin{aligned} E[\hat{X}^N] &= \frac{1}{2N+1} \sum_{n=-N}^N m_x = m_x \\ E[r_{xx}^N(m)] &= \frac{1}{2N+1} \sum_{n=-N}^N \gamma_{xx}(m) = \gamma_{xx}(m) \end{aligned}$$

Vegem el cas de la variància de la mitjana. Aplicant que $Var[X] = E[X^2] - E[X]^2$

$$\begin{aligned} E[(\hat{X}^N)^2] &= E \left[\left(\frac{1}{2N+1} \sum_{n=-N}^N X_n \right) \left(\frac{1}{2N+1} \sum_{n=-N}^N X_n \right) \right] = \\ &= \frac{1}{(2N+1)^2} \sum_{m=-N}^N \sum_{n=-N}^N E[X_m X_n] = \frac{1}{(2N+1)^2} \sum_{m=-N}^N \sum_{n=-N}^N \gamma_{xx}(m-n) = \\ &= \frac{1}{2N+1} \sum_{m=-2N}^{2N} \left(1 - \frac{|m|}{2N+1} \right) \gamma_{xx}(m) \end{aligned}$$

obtenim

$$Var[\hat{X}^N] = \frac{1}{2N+1} \sum_{m=-2N}^{2N} \left(1 - \frac{|m|}{2N+1} \right) c_{xx}(m)$$

De forma que vegem que una condició per que aquesta successió convergisca a zero és

$$\sum_{-\infty}^{\infty} |c_{xx}(m)| < \infty$$

En general assumim que els processos que tractem són ergòdics respecte a la mitjana i l'autocorrelació, de forma que és suficient amb tenir una mostra.

Podem tractar també amb l'autocorrelació en l'espai de freqüències a través del següent teorema

Teorema 3.2.1 (Wiener-Khinchin). *Siga x_n una sèrie temporal estacionària, amb funció d'autocorrelació absolutament sumable*

$$\sum_{h=-\infty}^{\infty} |\gamma_{xx}(h)| < \infty$$

obtenim la densitat espectral de potència (PSD) com la seua transformada de Fourier

$$\Gamma_{xx}(f) = \sum_{m=-\infty}^{\infty} \gamma_{xx}(m) e^{-i2\pi fm}$$

i amb la transformada inversa

$$\gamma_{xx}(m) = \int_{-\frac{1}{2}}^{\frac{1}{2}} \Gamma_{xx}(f) e^{i2\pi fm} df$$

Notem que

$$E[X_n^2] = \gamma_{xx}(0) = \int_{-\frac{1}{2}}^{\frac{1}{2}} \Gamma_{xx}(f) df$$

doncs podem considerar al PSD com la distribució de la potència en funció de la freqüència.

Tant amb la autocorrelació com la PSD són funcions que caracteritzen el soroll estacionari. Aquestes però no son conegudes, i per tant utilitzarem enllloc estimadors del PSD. Definim també dos tipus de soroll que utilitzarem més endavant.

Definició 3.2.6. Una sèrie temporal és soroll Gaussià si la distribució de probabilitat conjunta segueix una distribució normal multivariable

$$P(X) = \frac{1}{\det(2\pi C)^{1/2}} \exp \left[-\frac{1}{2} \sum_{i,j} (X_i - \mu)(X_j - \mu) C_{ij}^{-1} \right]$$

on C_{ij}^{-1} és la inversa de la matriu de covariància en i, j .

Definició 3.2.7. Direm que el soroll és blanc si $c_{xx}(i, j) = \delta_{ij}\sigma^2$ i $\Gamma_{xx}(f) = \sigma^2$.

3.3. Blanqueig de soroll

El soroll dels detectors de LIGO-Virgo és normalment estacionari i Gaussià [37], de forma que les freqüències $\tilde{x}(k)$ no estan correlacionades entre si i cadascuna segueix una distribució Gaussiana amb fase aleatoria i amplitud $\Gamma_{xx}(f)^{\frac{1}{2}}$. Una tècnica molt convenient és equalitzar l'amplitud de totes les freqüències a través del blanqueig de soroll.

Una forma molt usual de realitzar aquest procés és dividir cada valor de la transformada de Fourier discreta per la corresponent amplitud de densitat espectral $\Gamma_{xx}(f)^{\frac{1}{2}}$ i tornar a l'espai del temps. Per començar aquest procés cal calcular la transformada de Fourier i per fer-ho eficientment es gasta el mètode *FFT*. Aquest mètode però, assumeix periodicitat sobre les dades, de forma que cal aplicar una funció de finestra per obtindre una sèrie temporal periòdica que mantinga el màxim d'informació possible. A continuació estimem l'amplitud espectral i obtenim els nous valors del *DFT* $\tilde{X}_w(f) = \frac{X(f)}{\Gamma_n^{1/2}(f)}$. Per últim podem aplicar la transformada inversa per obtindre la sèrie temporal blanquejada.

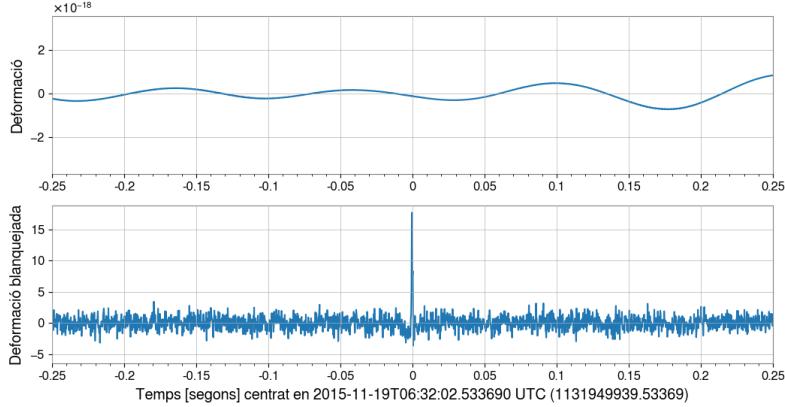


Figura 3.5: Exemple de blanquejament d'un segment de deformació mesurada en Ligo-Hanford centrada en un *glitch* de tipus Blip. Dalt es mostra la deformació mesurada pel detector i baix el resultat d'aplicar un blanquejament de soroll, on si que s'aprecia l'excés de potència del *glitch*.

També és possible aplicar un blanqueig en l'espai de temps com en l'experiment que veurem en Secció 5.1.1. Es pot fer ajustant un model autoregressiu a la sèrie temporal.

Definició 3.3.1. Siga un procés estocàstic $\{X_t, t \in \mathbb{Z}\}$, direm que és un procés autoregressiu d'ordre p , denotat per $AR(p)$ si es pot representar com

$$X_t = \phi_1 X_{t-1} + \dots + \phi_p X_{t-p} + Z_t + c$$

on $Z_t \sim WN(0, \sigma^2)$ és un soroll blanc i $Cov[Z_t, X_{t-s}] = 0$ per a tot $s \geq 1$.

Existeixen molts mètodes per ajustar un model $AR(p)$ a una sèrie temporal, doncs una volta trobat aquest, podem filtrar el soroll blanc Z_t per blanquejar les dades.

3.4. Estimació del *PSD*

Per estimar el *PSD*, podem utilitzar diferents estadístics en què s'intercanvia resolució en l'espai de freqüències per reduir la seua variància. En primer lloc definim el periodograma

$$P_{xx}(f) = \frac{1}{N} |\tilde{x}(f)|^2$$

Aquest és un estimador asimptòticament imparcial, però la seua mitjana és una convolució de la vertadera *PSD* i la variància no convergeix a 0 amb N . Tot i que no és un estimador consistent, és un element utilitzat en altres estimadors del *PSD*.

El mètode de Bartlett consisteix en subdividir la sèrie temporal en K trossos sense intersecció de la mateixa mida M .

$$x_i(n) = x(n + iM)$$

amb $i = 0, 1, \dots, K - 1$, $n = 0, 1, \dots, M - 1$.

Per a cada tros calculem el corresponent periodograma

$$P_{xx}^i(f) = \frac{1}{M} \left| \sum_{n=0}^{M-1} x_i(n) e^{-i2\pi f n} \right|^2, \quad i = 1, \dots, K - 1$$

i obtenim el *PSD* com a la mitjana dels periodogrames

$$P_{xx}^B(f) = \frac{1}{K} \sum_{i=0}^{K-1} S_x^i(f)$$

El mètode de Welch és una modificació del mètode de Bartlett en que permetem un solapament entre els trossos, usant una funció de finestra a l'hora de calcular els respectius periodogrames per compensar la sobrerepresentació.

3.5. Q-transformació

La Q-transformació (*Q-transform*) és una alternativa al *DFT* per analitzar les freqüències d'una senyal. Quan calculem una *DFT*, calculem les transformades de Fourier per a freqüències donades per $f_k = k \frac{F_s}{N}$ per a $0 \leq k < N$, on F_s és la freqüència de mostreig i N el número de mostres de que disposem. L'interval entre cadascuna s'anomena la resolució o *bandwidth* d'una freqüència, i es defineix per $\Delta_f = f_{k+1} - f_k$, en aquest cas $\Delta_f^{DFT} = \frac{F_s}{N}$. Una mesura interessant aleshores és el ratio de freqüència a resolució, que en la *DFT* és

$$\frac{f_k}{\Delta_f^{DFT}} = k,$$

és a dir incrementa proporcionalment a la freqüència.

Açò no és sempre eficient però, en la música per exemple, distingim una nota d'una altra a una distància d'un semitó, que equival a un 6% respecte de la nota original. Si volem distingir un La3 = 220Hz aleshores un semitó per damunt o per baix s'obté amb el producte per $2^{\frac{1}{12}} \approx 1,05946$ (assumint un temperament igual, això és dividir cada octava en 12 notes a freqüències equidistants). Així cal una resolució d'uns 13Hz per poder distingir-ho d'un La3 o La#3, en canvi fent càlculs similars, per distingir un La5 = 880Hz cal una resolució d'uns 52Hz. Resulta, per tant, inefficient utilitzar un *DFT* en que per a freqüències més altes estem calculant termes de freqüències en excés. És la naturalesa proporcional dels intervals musicals que dona lloc a aquest comportament, i ens pot dur a desenvolupar un mètode alternatiu en que el ratio de freqüència a resolució siga constant, que pot ser aplicat a molts altres tipus de dades.

D'aquesta forma en [7] es desenvolupa el mètode de la *Q-transform* per resoldre aquest problema i permetent identificar notes musicals de forma superior a *DFTs*. Aquest mètode és aplicat per obtindre espectrogrames d'ones gravitacionals també. El mètode del *DFT* presenta diversos problemes, si triem una finestra de temps per

analitzar dades, la duració d'aquesta determina una resolució de freqüències constant. Per freqüències baixes és necessària una llarga duració per acumular suficients termes de freqüències, mentres que per a altes freqüències es preferible gastar curtes duracions per millor resolució temporal.

Un altre mètode per estudiar senyals de curta duració en l'espai de freqüències és a través dels *wavelet transforms*, un altre tipus de descomposició de senyals en que s'utilitza una funció com a base o *wavelet* i es construeix una base a partir de contraccions, expansions i traslació d'aquesta. Per exemple en [15] utilitzen aquests dos mètodes per construir algorismes de detecció de events gravitacionals molt curts amb molta efectivitat. En els experiments que realitzarem però, només farem ús del *PSD* i del mètode de la *Q-transform*.

Per construir el mètode, en primer lloc triem una freqüència base f_0 que no té perquè ser 0 (amb el qual perdrem la possibilitat de definir una transformada inversa) i triem cada successiva freqüència com a producte de l'anterior per una potència de $2^{\frac{1}{b}}$ per convenció, $f_k = f_0 2^{\frac{k}{b}}$. D'aquesta forma

$$\Delta_k^{cq} = f_k (2^{\frac{1}{b}} - 1)$$

i així com voliem obtenim un ratio constant

$$Q := \frac{f_k}{\Delta_k^{cq}} = \left(2^{\frac{1}{b}} - 1\right)^{-1}.$$

Per definir una transformada amb aquests propietats, partim de la transformada de Fourier

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-2\pi i n k / N}$$

i notem que si triem $N := N_k = \frac{F_s}{\Delta_k^{cq}} = Q \frac{F_s}{f_k}$ aleshores la resolució corresponent seria $\Delta_k^{DFT} = \frac{F_s}{N_k} = \frac{f_k}{Q}$ i per tant $\frac{f_k}{\Delta_k^{DFT}} = Q$. Com que també $\frac{f_k}{\Delta_k^{DFT}} = k$ cal triar $k = Q$. Notem que si $Q \in \mathbb{Z}$ aquesta transformació és equivalent a la *DFT* de la freqüència Q amb finestra de duració $Q \frac{F_s}{f_k}$.

Així, definim la constant *Q-transform* amb freqüència base f_0 i freqüència màxima f_{max} com

$$X^{cq}(k) = \frac{1}{N_k} \sum_{n=0}^{N_k-1} x(n) w_{N_k}(n) e^{-2\pi i n Q / N_k}, \quad 0 \leq k < K$$

on $N_k = \lceil Q \frac{F_s}{f_k} \rceil$, $K = b \log_2 \left(\frac{f_{max}}{f_0} \right)$, $Q = \left(2^{\frac{1}{b}} - 1\right)^{-1}$ i w_N és alguna funció finestra utilitzada per reduir la fuga espectral.

3.6. Espectrogrames

El contingut de freqüències d'una senyal pot variar amb el temps i per analitzar aquest canvi es gasten espectrogrames. Un espectrograma representa una funció espectral en un interval de temps, de forma que a cada instant en el temps anem desplaçant aquest interval per apreciar canvis en la distribució de freqüències. Per fer-ho, es selecciona la duració de cada interval i si hi ha solapament o no entre aquests. Funcions típiques mostrades en espectrogrames són el *PSD*, *ASD* o a la *constant Q-transform*. Per exemple en (3.6) mostrem un espectrograma en que a cada instant mostrem els *ASD* i *constant Q-transform* d'un *glitch* de tipus *Blip*.

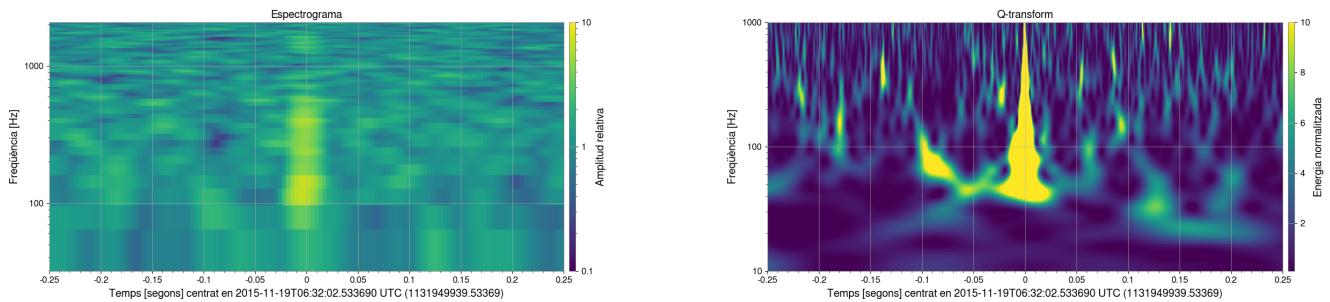


Figura 3.6: Espectrogrames del *glitch* mostrat en (3.5). A l'esquerra es tracta d'un espectrograma construït amb la *FFT* i a la dreta amb la *constant Q-transform*. Per mostrar amb claredat s'ha aplicat filtres de blanquejament i diversos processos de neteja. El mapa de colors és logarítmic en la gràfica esquerra i en ambdós casos els valors han sigut normalitzats.

Capítol 4

Machine learning

4.1. Fonaments

El camp de l'aprenentatge automàtic (*machine learning*) tracta de modelar el comportament d'un conjunt de dades, ja siga per donar prediccions futures, classificar-les o entendre millor la seua estructura. Aquest no és un camp nou, però si que ho és l'enorme increment del poder de computació en les últimes dècades. L'avanç tecnològic ha permés l'ús de models complexes i del seu entrenament amb grans quantitats de dades, que al seu temps ha incitat al desenvolupament de models més adients i especialitzats a les diferents tasques. La seua aplicació en la visió artificial s'ha desenvolupat amb molta rapidesa des de les primeres xarxes neuronals convolucionals modernes [28], aconseguint excel·lent resultats en tasques de classificació d'imatges. L'objectiu d'aquesta secció serà donar una breu introducció al funcionament d'una xarxa neuronal, el seu procés d'entrenament en tasques de classificació i la intuïció i implementació d'una xarxa neuronal convolucional (*CNN*).

Siga un conjunt de dades $\mathcal{D} = \{(x^{(i)}, y^{(i)}) \in \mathbb{R}^d \times \mathbb{R}^q\}_{i=1}^n$ on $x^{(i)} \in \mathbb{R}^d$ són variables independents i $y^{(i)} \in \mathbb{R}^q$ les variables dependents o etiquetes. Considerem un model $h_\theta(x)$ definit per paràmetres θ i plantegem la tasca d'optimització

$$\min_{\theta} J(\theta) = \min_{\theta} \frac{1}{n} \sum_{i=1}^n J^{(i)}(\theta)$$

per a una funció de cost J definida sobre cada valor $i = 1, \dots, n$.

Aquest és el plantejament fonamental de tota tasca que volem resoldre amb aprenentatge automàtic, ens podem fixar que hi ha molts components per especificar i que és essencial triar amb cura per obtindre bons resultats. Tot algorisme d'optimització té un biaix inductiu per a cert classe de problemes d'optimització [10], doncs aquesta decisió és essencial. Per enfocar-se a una tasca d'aprenentatge automàtic cal:

- Triar un model adient per a modelar les dades.
- Triar una funció de cost, de forma que la seua optimització s'alinee amb l'objectiu de la tasca.
- Triar un mètode d'optimització trobant un equilibri entre efectivitat i eficiència.
- Avaluar el model entrenat, comprovant la seua capacitat de generalització sobre les dades.

A l'hora de triar el model, és important garantir la seua capacitat d'aprendre una gran varietat de possibles funcions. En un principi no coneixem la classe de funcions a la qual pertany el model òptim. És, per tant,

preferible que tinga una gran expressivitat. En general, volem un model que puga aproximar qualsevol funció contínua, és a dir, que complisca un cert teorema d'aproximació universal.

El tipus de xarxa neuronal més simple amb aquesta propietat és la multicapa de perceptrons (*MLP*) [3]. Aquest arreplega l'essència de les xarxes neuronals més modernes: la composició de funcions simples per formar una funció més complexa de forma modular. En aquest cas la funció simple es tracta d'un perceptró, que consisteix en una composició d'una funció afí i una funció no lineal σ que anomenem funció d'activació

$$f_{w,b}(x) = \sigma(w^T x + b)$$

Aquestes funcions les anomenem neurones i les disposem en una sèrie de capes de forma que reben unes dades d'entrada i envien el seu resultat a cada funció de la següent capa successivament fins a obtindre uns valors d'eixida. Més endavant quan expliquem el seu entrenament formalitzarem la seua estructura.

L'elecció de la funció de cost depén de la tasca, i en aquest cas considerarem la tasca de classificació. En una tasca de classificació les etiquetes representen variables categòriques o classes, i l'objectiu del model és determinar a quina classe pertany un valor x .

Per representar les classes podem gastar la codificació de *one-hot encoding*, de forma que si hi ha q classes $y \in \{1, 2, \dots, q\}$ les representariem per un vector $y = (0, \dots, 0, 1, 0, \dots, 0) \in \mathbb{R}^q$ en que la posició del 1 representa la classe. D'aquesta forma, sembla raonable modelar el vector probabilitat que un punt x pertany a cada classe. Per garantir que els valors d'eixida segueixen una distribució de probabilitat, és comú aplicar una funció softmax al resultat de l'última capa $O = (o_1, \dots, o_q)$,

$$\text{softmax}(O) = \left(\frac{e^{o_1}}{\sum_{n=1}^q e^{o_n}}, \dots, \frac{e^{o_q}}{\sum_{n=1}^q e^{o_n}} \right)$$

de forma que $h_\theta(x) = \text{softmax}(O)$ compleix que els seus components són positius,

$$\sum_{i=1}^q \frac{e^{o_i}}{\sum_{n=1}^q e^{o_n}} = \frac{\sum_{i=1}^q e^{o_i}}{\sum_{n=1}^q e^{o_n}} = 1$$

i a més a més com que l'exponencial és monòtona, el valor més gran de O correspon a la probabilitat més gran.

Ara com a funció de cost podem gastar l'entropia encreuada (*cross-entropy loss*). Per a una predicció $h_\theta(x)$ i l'etiqueta correcta y , definim la funció de cost d'una sola predicció com

$$J(y, h_\theta(x)) = - \sum_{j=1}^q y_j \log(h_\theta(x)_j).$$

Des del punt de vista de la teoria de la informació, la funció $J(y, \hat{y})$ representa la sorpresa esperada d'observar dades provenint de la distribució real y assumint que segueixen la distribució \hat{y} [13]. Així minimitzar aquesta funció implica apropar la probabilitat obtinguda a la real.

Ara que tenim el model i la funció de cost d'una classificació, cal plantejar un mètode per entrenar el model. Per realitzar l'optimització utilitzarem l'algorisme del descens de gradient (*Gradient Descent*), que consisteix a calcular millors aproximacions dels paràmetres de forma recursiva fins la convergència de $J(\theta)$, seguint a cada pas la direcció de màxima minimització de $J(\theta)$ per als paràmetres

$$\theta := \theta - \alpha \nabla_\theta J(\theta).$$

La constant α es denomina la velocitat d'aprenentatge. Aquest algorisme requereix iterar sobre tot el conjunt de dades a cada pas, que pot resultar massa costós. Per reduir el cost computacional, se sol fer servir el descens

de gradient estocàstic per subgrups (*mini-batch SGD*), en què a cada pas només s'optimitza en la direcció de màxima minimització d'un subconjunt de les dades I

$$\theta := \theta - \frac{\alpha}{|I|} \sum_{i \in I} J^{(i)}(\theta).$$

Per aplicar *SGD*, seleccionarem una mida $|I| = k$ per a aquests subgrups, dividirem el conjunt \mathcal{D} de forma que els subconjunts tinguin k elements (excepte un últim subgrup incomplet si $|\mathcal{D}|$ no és divisible per k). A més a més, entrenarem el model múltiples voltes sobre el mateix conjunt de dades, i cada passada sobre les dades l'anomenarem una època.

4.1.1. Multicapa de perceptrons

Finalment, cal formalitzar el model *MLP*. Aquest està format per tres tipus de capes: la capa d'entrada, les capes amagades i la capa d'eixida. El denotem com a una funció h_θ en que θ representa tots els paràmetres de cada neurona de cada capa.

Introduïm primer un exemple senzill amb una capa amagada. Siga un valor del conjunt de dades $x \in \mathbb{R}^d$. Aleshores tenim d neurones en la capa inicial, que són les components del valor introduït. Si la capa amagada té h neurones, representem cadascuna per una funció $H_i(x) = \sigma(w_i^{(1)T}x + b_i)$ on $w_i^{(1)} \in \mathbb{R}^d$, $b_i \in \mathbb{R}$ són els paràmetres corresponents a la transformació afí.

A més a més, podem representar la capa en notació vectorial si definim $H : \mathbb{R}^d \rightarrow \mathbb{R}^h$.

$$H(x) = \sigma(W^{(1)T}x + B^{(1)}) \in \mathbb{R}^h.$$

on $W^{(1)} = \begin{pmatrix} w_1^{(1)} & \dots & w_h^{(1)} \end{pmatrix} \in \mathbb{R}^{d \times h}$ i $B_1 = \begin{pmatrix} b_1 \\ \vdots \\ b_h \end{pmatrix} \in \mathbb{R}^h$ i on σ actua sobre cada element del vector.

Seguint aquesta notació, la capa d'eixida és una funció $O : \mathbb{R}^h \rightarrow \mathbb{R}^q$ sense funció d'activació

$$O(x) = W^{(2)T}H(x) + B^{(2)}$$

amb paràmetres $\begin{pmatrix} w_1^{(2)} & \dots & w_q^{(2)} \end{pmatrix} = W^{(2)} \in \mathbb{R}^{h \times q}$ i l'esbiaix $B^{(2)} \in \mathbb{R}^q$ que dona el resultat final del model $h_\theta(x) = O(x)$.

La notació vectorial també es pot extender a la introducció de grups de dades al model, que és important per a realitzar descens en gradient estocàstic per subgrups. Si introduïm un grup de dades $(x^{(1)} \ \dots \ x^{(m)}) = X \in \mathbb{R}^{d \times m}$, aleshores podem considerar la capa amagada com una funció $H : \mathbb{R}^{d \times m} \rightarrow \mathbb{R}^{h \times m}$ i denotem

$$H(X) = \sigma(W^{(1)T}X + B^{(1)})$$

on σ actua sobre cada element de la matriu i $B^{(1)}$ es suma a cada columna de $W^{(1)T}X$. D'aquesta forma la columna i de $H(X)$ correspon a $H(x^{(i)})$. De forma anàlega definiriem la capa d'eixida no incloent la funció d'activació.

Suposem ara que volem definir k capes internes al model cadascuna amb h_i neurones per a $1 \leq i \leq k$ i volem

classificar m classes (definim $h_0 = d, h_{k+1} = m$). Aleshores podem escriure $h_\theta(x)$ per a $x \in \mathbb{R}^d$ com

$$\begin{aligned} H^{(1)} &= \sigma(W^{(1)T}x + B^{(1)}) = \sigma(z^{(1)}) \\ H^{(2)} &= \sigma(W^{(2)T}H^{(1)} + B^{(2)}) = \sigma(z^{(2)}) \\ &\vdots \\ H^{(k)} &= \sigma(W^{(k)T}H^{(k-1)} + B^{(k)}) = \sigma(z^{(k)}) \\ H^{(k+1)} &= W^{(k+1)T}H^{(k)} + B^{(k+1)} = z^{(k+1)} \\ O &= \text{softmax}(H^{(k+1)}) \\ L &= J(y, O) \end{aligned}$$

on $B^{(i)} \in \mathbb{R}^{h_i}, W^{(i)} \in \mathbb{R}^{h_{i-1} \times h_i}$ per a $1 \leq i \leq k+1$.

Per calcular el gradient de la funció de cost utilitzem l'algorisme de retropropagació, aprofitant l'estructura repetitiva del model per aplicar la regla de la cadena a cada composició de funcions i reaprofitant tot càcul repetit.

L'objectiu és calcular la derivada parcial de L respecte de cada paràmetre per obtindre el gradient. Per simplificar càculs gastem les derivades totals, que és equivalent a treballar en derivades parcials en notació vectorial.

Definició 4.1.1. Siga una funció $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$ derivable, definim la derivada total respecte de les seues m variables $x = (x_1, \dots, x_m)$ com

$$\frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_m} \end{pmatrix}$$

Ara apliquem la regla de la cadena a cada composició de funcions des de l'última capa fins la primera. Siga $1 \leq r < k+1$, aleshores com $H^{(r)} = \sigma(W^{(r)T}H^{(r-1)} + B^{(r)}) = \sigma(z^{(r)})$, aplicant la regla de la cadena amb la variable $z^{(r)}$ (que en derivades parcials és simplement aplicar-la a cada derivada parcial)

$$\begin{aligned} \frac{\partial L}{\partial B^{(r)}} &= \frac{\partial L}{\partial z^{(r)}} \frac{\partial z^{(r)}}{\partial B^{(r)}} = \frac{\partial L}{\partial z^{(r)}} \\ \frac{\partial L}{\partial W^{(r)}} &= \frac{\partial L}{\partial z^{(r)}} \frac{\partial z^{(r)}}{\partial W^{(r)}} = \frac{\partial L}{\partial z^{(r)}} H^{(r-1)}. \end{aligned}$$

Doncs es prou amb calcular $\delta^{(k)} = \frac{\partial L}{\partial z^{(r)}}$. Ho calcularem de forma inductiva, de $r = k+1$ a $r = 1$. Si $r = k+1$,

$$\delta^{(k+1)} = \frac{\partial L}{\partial H^{(k+1)}} = \frac{\partial J(y, \text{softmax}(H^{(k+1)}))}{\partial H^{(k+1)}} = (O - y).$$

Si $r < k+1$,

$$\delta^{(r)} = \frac{\partial L}{\partial z^{(r)}} = \frac{\partial L}{\partial H^{(r)}} \frac{\partial H^{(r)}}{\partial z^{(r)}} = \frac{\partial L}{\partial H^{(r)}} \sigma'(z^{(r)}).$$

Ara com $z^{(r+1)} = W^{(r+1)T}H^{(r)} + B^{(r+1)}$,

$$\delta^{(r)} = \frac{\partial L}{\partial z^{(r+1)}} \frac{\partial z^{(r+1)}}{\partial H^{(r)}} \sigma'(z^{(r)}) = \frac{\partial L}{\partial z^{(r+1)}} W^{(r+1)T} \sigma'(z^{(r)}) = \delta^{(r+1)} W^{(r+1)T} \sigma'(z^{(r)})$$

doncs obtenim un algorisme per calcular totes les derivades $\frac{\partial L}{\partial \theta_i}$, implementat en pseudocodi en (7).

Algorisme 1: Retropropagació d'un model MLP amb softmax i entropia encreuada.

```

1 Calcular i guardar els valors de  $H^{(i)}, z^{(i)}, \forall i$  (passada cap endavant);
2  $\delta^{(k+1)} \leftarrow O - y;$ 
3 per  $r = k, \dots, 1$  (passada cap enrere) fer
4    $\delta^{(r)} \leftarrow \delta^{(r+1)} W^{(r+1)T} \sigma'(z^{(r)})$ ;
5    $\frac{\partial L}{\partial B^{(r)}} \leftarrow \delta^{(r)}$ ;
6    $\frac{\partial L}{\partial W^{(r)}} \leftarrow \delta^{(r)} H^{(r-1)}$ ;
7 fi

```

4.1.2. Regularització

Els models de *machine learning* com les multicapes de perceptrons són models molt expressius i poden modelar variables amb comportaments molt complexes amb suficients mostres d'aquestes. En la pràctica però, disposem d'un conjunt de dades finit, que no representa el comportament total de les variables i que pot contindre soroll o falsos comportaments degut a la seua naturalessa finita. En aquests casos, sovint, una xarxa neuronal optimitza l'error sobre el conjunt de dades d'entrenament, però no sobre el conjunt de dades total, esdeveniment que anomenem *overfitting*. De fet, la majoria de les xarxes neuronals tenen la suficient complexitat per a memoritzar un conjunt de dades de la grandària típica de la qual disposem (per exemple de 10000 mostres) perdent tota capacitat de generalitzar. Tot i que tenen aquesta capacitat, normalment una xarxa neuronal entrenada amb *SGD* sol arribar a un estat òptim durant l'entrenament que generalitza les dades, potser al aprendre estadístiques incrementalment complexes de les dades [44]. Quan aquest no és el cas, altres tècniques poden ser utilitzades per penalitzar o alentir l'entrenament de forma que la xarxa no represente en excés les dades d'entrenament.

Una tècnica molt simple però útil és parar l'entrenament quan la funció de cost sobre dades d'avaluació (sobre les quals no s'entrena) deixa de ser minimitzada (*early stopping*). Altres tècniques molt populars són l'adició de la norma dels paràmetres de les neurones a la funció de cost, de forma que la complexitat del model es restringeix i a voltes és efectiu. Segons el tipus de norma que s'utilitza anomenem regularització l_2 o regularització l_1 .

En aquest cas expliquem amb detall la regularització amb l'ús del *dropout* [20], que resulta molt efectiva en els models aplicats als experiments. Una forma de reduir l'*overfitting* es combinar les prediccions de diversos models. Si disposem d'un conjunt de dades \mathcal{D} , i una sèrie de models \mathcal{M}_i que defineixen una distribució de probabilitat sobre aquest aleshores donat un valor x del qual volem predir un valor associat, podem obtindre la predicción a partir de la probabilitat mixta:

$$p(t|x, \mathcal{D}) = \sum_i P(t|x, \mathcal{D}, \mathcal{M}_i) P(\mathcal{M}_i|x, \mathcal{D}) = \sum_i P(t|x, \mathcal{D}, \mathcal{M}_i) P(\mathcal{M}_i|\mathcal{D})$$

Açò és equivalent a calcular la predicción mitjana dels models, assumint que cada model té una probabilitat a posteriori $P(\mathcal{M}_i|\mathcal{D}) \propto P(\mathcal{M}_i)P(\mathcal{D}|\mathcal{M}_i)$ definida per una probabilitat a priori $P(\mathcal{M}_i)$ i l'evidència del model $P(\mathcal{D}|\mathcal{M}_i)$. Aquesta estratègia pot evitar eficaçment l'*overfitting* si algun dels models modela correctament les dades [16].

Aquests valors poden ser calculats amb suficients suposicions, però el principal limitant en la pràctica és l'impossibilitat d'entrenar un gran nombre de models. La idea presentada en l'article [20] és la de considerar múltiples models dins d'un model. Per fer-ho, eliminem diverses neurones de cada capa de forma aleatòria per obtenir diferents models (4.1). En cada època, assignem a cada neurona una probabilitat p de participar al model de forma que el resultat d'una capa amagada H amb *dropout* seria

$$\hat{H}_i = \begin{cases} 0 & \text{amb probabilitat } 1-p \\ H_i & \text{amb probabilitat } p \end{cases}$$

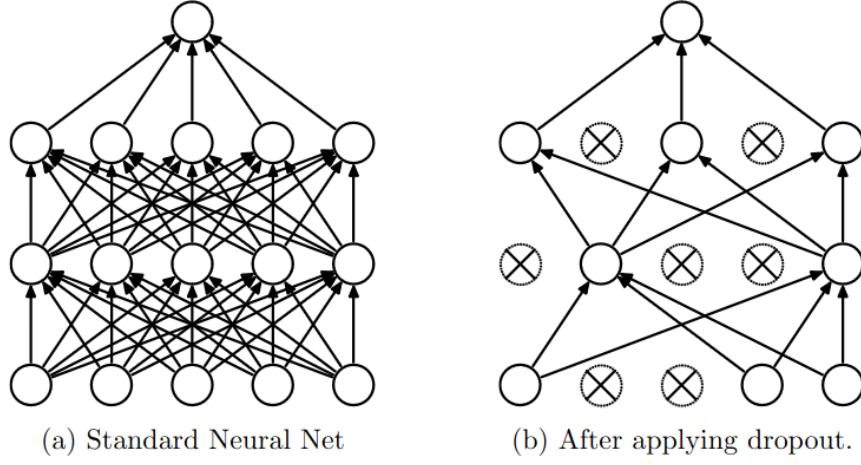


Figura 4.1: Diagrama d'una capa amagada amb dropout.

on \hat{H}_i és el nou valor de la neurona i de la capa H . A l'hora d'evaluar el model combinació d'aquests, podem aproximar la mitjana de les prediccions utilitzant el model sense *dropout* i escalant cada neurona per p de forma que el valor esperat de cada neurona siga el mateix en l'entrenament i l'avaluació.

4.1.3. Avaluació i búsquedas d'hiperparàmetres

Per avaluar un model cal comprobar la seua capacitat de generalització. Si entrenem un model amb un conjunt de dades estem optimitzant la funció objectiu sobre aquest. Una bona pràctica és reservar una part del conjunt de dades sobre la que no entrenem i evaluar la funció objectiu sobre aquesta a posteriori.

Podem utilitzar l'avaluació del model per trobar quina arquitectura, quina velocitat d'aprenentatge o qualsevol paràmetre que es manté fixe durant l'entrenament funciona millor. Aquests paràmetres els anomenem hiperparàmetres i en el procés d'optimitzar-los correm també el risc d'aplicar *overfitting* sobre el conjunt de dades d'avaluació. És per això que normalment es reserva una segona part del conjunt de dades original per testejar el model final.

4.2. Xarxes neuronals convolucionals

L'introducció al les xarxes neuronals convolucionals que es dona en aquesta secció és una adaptació i resum dels continguts del llibre digital [45].

Les xarxes neuronals convolucionals (*CNN*) tracten d'aprofitar més informació sobre l'estructura d'un tipus de dades en particular: imatges representades per matrius de píxels. Suposem que tenim una imatge $x \in \mathbb{R}^{w \times h}$ i considerem un model *MLP* amb una capa amagada. Denotem $H_{i,j}$ el resultat d'aquesta capa en rebre el píxel (i, j) i omitim per ara la funció d'activació, de forma que s'expressa com

$$H_{i,j} = \sum_{k,l} W_{i,j,k,l} x_{k,l} + B_{i,j}$$

per a k, l recorrent l'amplària i altura de l'imatge respectivament.

Aquesta funció però, tindria un nombre de paràmetres excessiu per a la grandària d'imatges usual. Podem utilitzar propietats de les imatges per deduir una funció més eficient:

1. Invariància de translació [5]: les propietats d'un tros d'imatge no depenen de quina regió d'aquesta es trobe.
2. Localitat: els píxels propers entre si haurien de semblar-se més que aquells llunyans

El primer punt implica que un píxel $x_{i,j}$ deuria tenir la mateixa representació $H_{i,j}$ si el desplaçarem a una altra posició de la imatge, és a dir que $H_{i,j}$ no depén de (i, j)

$$H_{i,j} = \sum_{k,l} W_{k,l} x_{k,l} + B$$

El segon punt ens fa pensar que la representació del píxel $H_{i,j}$ no té perquè dependre de tots els píxels de la imatge, sinó només d'un subconjunt de píxels proper a aquest $i - K \leq a \leq i + K, j - K \leq b \leq j + K$.

$$H_{i,j} = \sum_{a=-K}^K \sum_{b=-K}^K W_{a,b} x_{k+a,l+b} + B$$

Hem obtingut així un kernel $W_{a,b}$ que opera efectuant una operació de correlació-encreuada sobre les dades, d'on ve el nom de convolucional (tot i que no és correcte aquest terme). Cal incloure també una funció d'activació, per garantir la complexitat del model similarment al *MLP* i obtenim així la neurona d'una capa convolucional.

Quan formem una capa convolucional amb h neurones l'anomenem el nombre de canals d'eixida, ja que cadascuna produeix una matriu donant lloc a un tensor $\mathbb{R}^{w-K,h-K,h}$. Aquesta tercera dimensió s'anomena així perquè una imatge en color té per defecte tres canals (roig, blau i verd), doncs pot representar-se com un tensor $x \in \mathbb{R}^{w,h,3}$.

Ara si les capes convolucionals poden donar múltiples canals d'eixida, també tenen que admetre múltiples canals d'entrada. Per fer-ho, cal afegir canals al kernel, aplicar l'operació de correlació-encreuada i sumar els canals component a component per obtindre un canal per kernel. Si $x \in \mathbb{R}^{w,h,c}$, aleshores

$$H_{i,j} = \sum_{q=1}^c \sum_{a=-K}^K \sum_{b=-K}^K W_{a,b,q} x_{k+a,l+b,q} + B$$

i si incloem múltiples canals d'eixida podem denotar el corresponent a cada kernel afegint un índex més

$$H_{i,j,d} = \sum_{q=1}^c \sum_{a=-K}^K \sum_{b=-K}^K W_{a,b,q,d} x_{k+a,l+b,q} + B$$

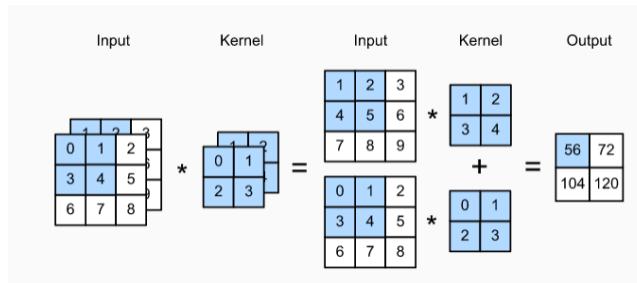


Figura 4.2: Exemple de capes convolucionals amb múltiples canals d'entrada. Imatge extreta de ([45] Ch. 7.4).

Altres paràmetres que a sovint s'utilitzen en les capes convolucionals són el *padding* i *stride*. Notem que en aplicar una capa convolucional amb kernel 5×5 a una imatge de dimensions 28×28 obtenim una imatge reduïda de 23×23 . Per mantindre el tamant de l'imatge i poder incloure més capes convolucionals, una opció és afegir zeros enmarcant l'imatge de forma que es mantinguin les dimensions. Així si afegim dues línies de zeros envoltant l'imatge estem augmentant les seues dimensions a 32×32 de forma que en aplicar el kernel obtindriem 27×27 píxels. Al nombre de capes de zeros que utilitzem per envoltar l'imatge és el *padding*.

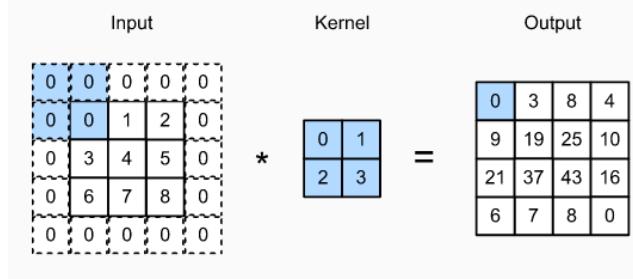


Figura 4.3: Exemple d'una capa convolucional amb kernel 2×2 i *padding* 1. Imatge extreta de ([45] Ch. 7.3).

Si per el contrari volem reduir més encara la dimensió de la imatge, en lloc d'aplicar el kernel a cada píxel podem triar un nombre, que anomenem *stride* i aplicar el kernel a intervals d'aquest valor. D'aquesta manera si apliquem un kernel 5×5 amb stride 2 sobre una imatge de 28×28 obtindriem una imatge reduïda de 12×12 .

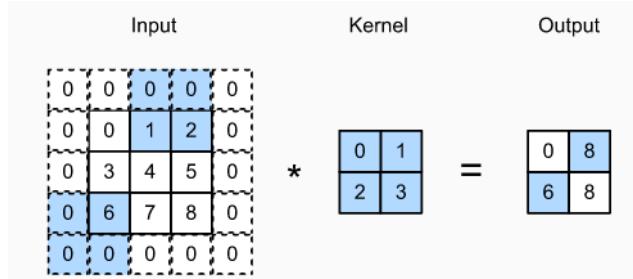


Figura 4.4: Exemple d'una capa convolucional amb kernel 2×2 i *stride* 2. Imatge extreta de ([45] Ch. 7.3).

Per construir una *CNN*, la forma més simple és incloure una sèrie de capes convolucionals al principi, condensant la informació de la imatge. A continuació convertim les matrius resultants en vectors i les introduïm en un *MLP*, per garantir l'aproximació universal. Aquest model es pot entrenar anàlogament al *MLP* a través de *SDG* amb retropropagació. L'algorisme de retropropagació és diferent per cada xarxa neuronal, però pot ser obtingut de forma anàlega al que hem deduït per al *MLP*. En els ordinadors aquest procés és realitzat de forma eficient a partir del graf computacional del model.

A sovint altre tipus de capes es combinades amb capes convolucionals per reduir la dimensió de les dades ràpidament amb poc esforç computacional al mateix temps que conservant la màxima informació. Les capes de normalització operen sobre l'input aplicant un kernel fixe (sense paràmetres) que calcula la mitjana o el màxim del seu camp receptiu. Normalment, és més efectiu aplicar el màxim, conformant el que anomenem una capa tipus *Max-Pool*, que la mitjana. Aquesta capa va ser per primera volta inclosa en una xarxa neuronal per al

reconeixement de paraules a partir de so [4] i proposat com a un mecanisme per al reconeixement d'imatges en el context de la neurociència cognitiva en [12].

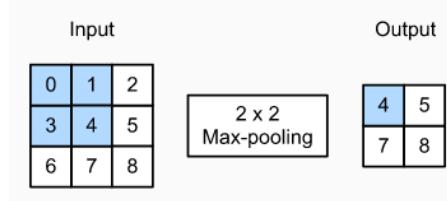


Figura 4.5: Exemple del l'aplicació d'un *Max-Pool* 2×2 a una matriu 3×3 . Imatge extreta de ([45] Ch. 7.5).

El primer exemple d'un model *CNN* exitós al camp del *computer vision* és el model *LeNet* [11], que va aconseguir resultats comparables als millors models del moment en la tasca de reconeixement d'imatges de díigits escrits a mà, el famós conjunt de dades *MNIST*. Aquest model utilitza *AvgPool* i activacions de tipus sigmoidal ($\sigma(x) = \frac{1}{1+e^{-x}}$). El primer model però que va demostrar que l'entrenament de funcions de kernel supera al seu disseny manual, va ser el model *AlexNet* [28]. Aquest va ser el guanyador d'una competició per classificar les imatges del conjunt de dades *ImageNet* [19] i té una profunditat i grandària major que el model *LeNet*. També incorpora capes tipus *MaxPool*, utilitza funcions d'activació *ReLU* ($ReLU(x) = x$ si $x > 0$ i 0 sinó), inclou *dropout* en les dos primeres capes totalment connectades. Podem comparar les seues arquitectures en la Figura 4.6, on vegem que es tracta de models molt similars en estructura i diferents en escala. El principal factor que va limitar l'aparició de *AlexNet* (van passar uns 17 anys des de l'aparició de *LeNet*) va ser computadores suficientment potents. El model *LeNet* va ser entrenat utilitzant *CPU* mentres que *AlexNet* va ser entrenat amb dos *GPs*, un tipus de processador optimitzat per al càlcul paral·lel d'operacions matricials.

El següent pas natural en la construcció de models *CNN* més complexos i amb més profunditat és l'abstracció dels seus components. En aquests models podem diferenciar una estructura clara, aplicar capes convolucionals per augmentar la complexitat, introduïr capes tipus *MaxPool* per reduir la dimensió i finalment afegir capes totalment connectades fins produir el classificador. Podem passar de pensar en capes a grups de capes o blocs convolucionals, aquesta idea es va iniciar amb l'anomenat *VGG* [22], en que es gasten blocs formats per una sèrie de capes convolucionals amb kernel 3×3 i *padding* 1 seguides d'una capa *MaxPool* amb kernel 2×2 i *stride* 2 de forma que cada bloc divideix en dos l'altura i amplaria de l'imatge que rep com a entrada 4.7. La idea principal però es la de parametrizar un model per el nombre de blocs i l'estructura d'aquests.

Per denotar de forma curta l'arquitectura d'un model per blocs utilitzarem una notació curta. Per denotar una capa convolucional amb kernel $k \times k$, n_o capes d'eixida i p *padding* escriurem (k, n_o, p) o (k, n_o) si no gastem *padding*. Així si comosem n_c capes convolucionals per formar un bloc com en Figura 4.7 podem denotar-les per $((k, n_{o,1}, p_1), \dots, (k_{n_c}, n_{o,n_c}, p_{n_c}))$ o $n_c \times (k, n_o, p)$ si cada capa és igual (omitint el nombre de capes si només inclou una). Si volem denotar una sèrie de blocs els escriuriem com a una llista de blocs seguint aquesta notació. D'aquesta manera podem parametrizar ràpidament un model només que definint els seus components genèrics com les capes de normalització, *dropout* o totalment connectades que conté.

Amb l'interés creixent en construir xarxes neuronals més profundes, van aparèixer el concepte de les capes de normalització per subgrups (*batch normalization*). Aquest adreça el fet que conforma augmenta la profunditat d'un model, en cada capa es transforma la distribució de les dades, el que s'anomena *covariance shift*. És preferible mantindre una mateixa distribució en una capa, ja que facilita el seu entrenament i evita problemes de desaparició de gradients amb certes funcions d'activació. Per fer-ho de forma eficient, en [21] es proposa aplicar una transformació de forma que cada component de les dades tinga mitjana zero i variància 1, estimant la mitjana i variància a partir del subgrup triat en aplicar *SDG* durant l'entrenament. A més a més s'inclouen dos paràmetres per poder reescalar i esbiaixar les dades, per no limitar l'expressivitat de les dades. La transfor-

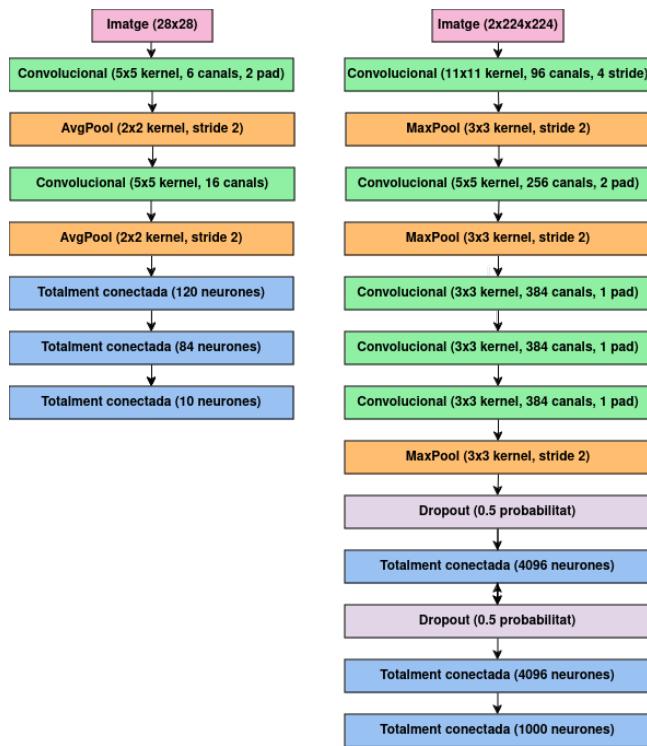


Figura 4.6: Arquitectura del model *LeNet* a l'esquerra i *AlexNet* a la dreta.

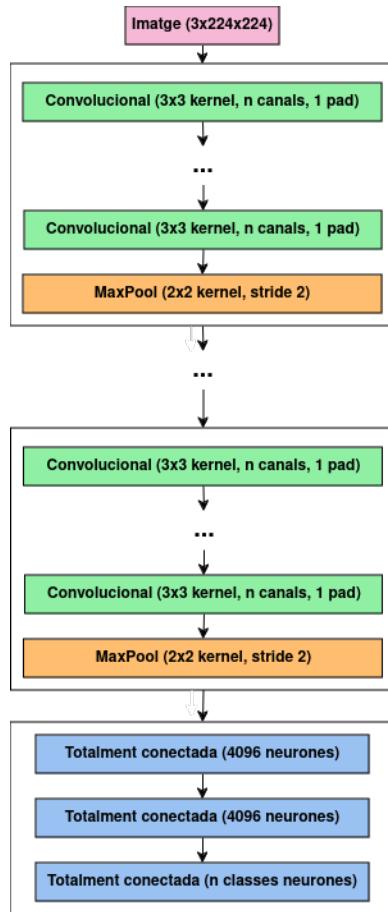


Figura 4.7: Arquitectura del model VGG.

mació proposada sobre un subgrup de m dades d'entrenament $\mathcal{B} = \{x_1, \dots, x_m\}$ seria la donada per la funció $BN_{\gamma,\beta}(x_i)$:

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i, \quad \sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \quad BN_{\gamma,\beta}(x_i) = \gamma \hat{x}_i + \beta$$

on $\epsilon > 0$ és un valor molt menut inclòs per evitar problemes numèrics.

Aquesta transformació pot ser inclosa com a una capa de normalització en una xarxa neuronal, i va ser un dels avanços que van donar a lloc al següent tipus de xarxes neuronals, les xarxes residuals (*Residual Nets*).

Les xarxes residuals reformulen cada capa convolucional com a una funció d'aprenentatge residual 4.8. Un dels principals factors de l'eficàcia d'una xarxa neuronal és la seua profunditat, una tendència clara als models estudiats fins a aquest punt. En construir un model més profund diversos problemes poden apareixer, com la desaparició/explosió de gradients (resolt per les capes de normalització) i en general major dificultat al seu aprenentatge com un decreixement en aquest no degut a sobreajust. La idea que dona lloc als models residuals *ResNet* [25] es la següent: si a un model menys profund, li afegim més capes i aquestes apliquen la funció identitat, aquest funcionarà tan bé o millor que l'original. La proposta es per tant construir un tipus de capa que sume el resultat de la capa anterior al seu resultat. Formalment, si volem que una capa modele una funció $\mathcal{H}(x)$, aleshores permetem que una part de la capa modele la funció $\mathcal{F}(x) = \mathcal{H}(x) - x$ i a aquesta li tornem a sumar la identitat $\mathcal{F}(x) + x$. S'hipotetitza així que és més fàcil entrenar una capa per ajustar-se al residu \mathcal{F} resultant de la capa anterior que a la funció completa \mathcal{H} i en el pitjor cas aquesta capa no modificarà al model anterior.

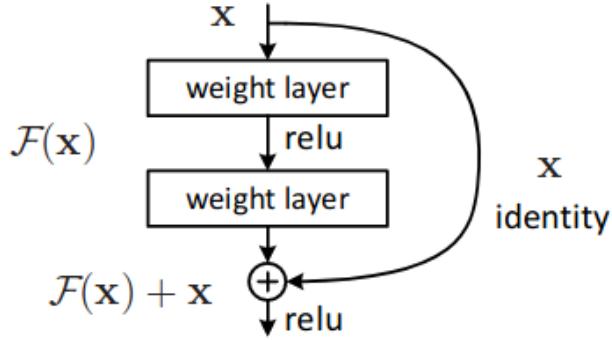


Figura 4.8: Esquema d'una capa residual, on es mostra com es relega les transformacions no lineals l'aprenentatge de $\mathcal{F}(x) + x$ a través de la inclusió d'un salt en la xarxa amb la funció identitat. Imatge extreta de l'article original sobre xarxes residuals [25].

Per adaptar aquesta idea a una capa convolucional, cal tenir en compte la transformació en el número de canals, amplària i altura de les dades d'eixida, ja que no li podrem sumar les dades d'entrada amb l'identitat. En aquest cas, projectarem les dades d'entrada a l'espai de les dades d'eixida a través d'una convolució amb kernel 1×1 i els valors de *stride* i canals d'eixida adients.

Com a exemple d'un model d'aquest tipus mostrem el model *ResNet18* 4.10, que té 18 capes dividides en una capa convolucional inicial, dues capes per cada bloc residual, 4 blocs residuals i una capa totalment connectada final.

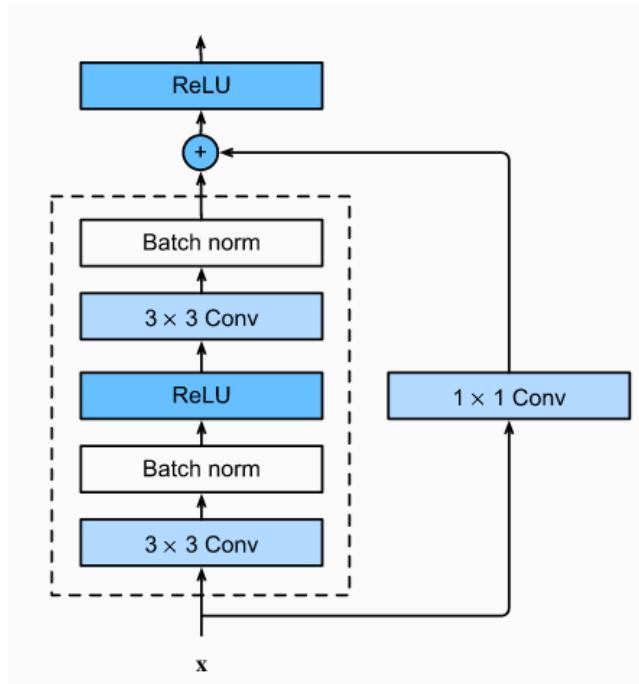


Figura 4.9: Estructura d'una capa convolucional residual. Imatge extreta de [45] Ch. 8.6.

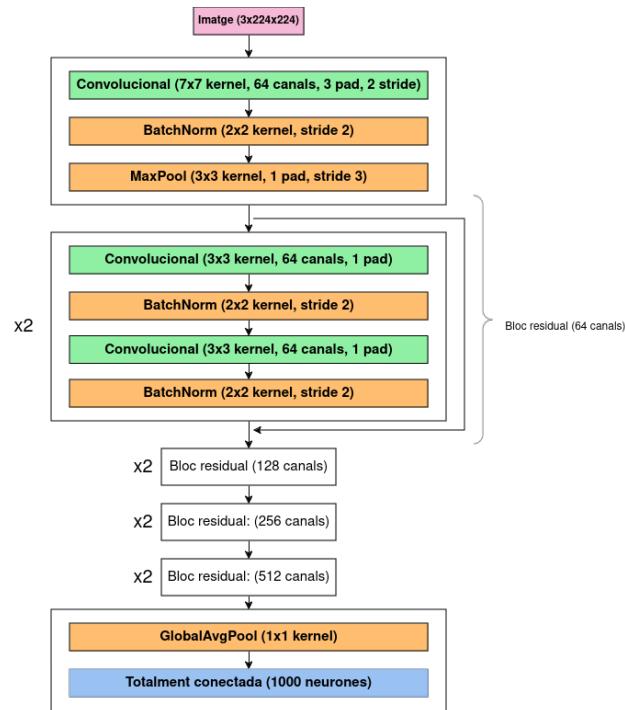


Figura 4.10: Arquitectura del model *ResNet18*.

Més enllà dels models vists fins ara, que donen resultats similars o superiors, es troben els models basats en l'estructura *transformer* adaptat per a tasques de *computer vision*, o els anomenats *Vision Transformer* (*ViT*). Tot i que en la secció experimental també inclourem alguns d'aquests models com a comparació, no els explicarem en aquesta secció, podem trobar una introducció en ([45] Ch. 11.8).

Capítol 5

Materials i Mètodes

5.1. Conjunts de dades

5.1.1. Espectrogrames de glitches sintètics

Abans de considerar *glitches* mesurats en detectors reals, podem tractar de simular-los i evaluar diversos models sobre aquestes dades, per asegurar una base sobre la qual treballar. Per fer-ho, utilitzarem el conjunt de dades generades i processades per a l'article [35] que han publicat¹ els seus autors de forma complementaria a aquest. De forma similar a aquest article, entrenarem diversos models per evaluar el seu funcionament. En aquest cas intentarem superar l'exactitud obtinguda pel model *CNN* proposat a l'article.

Aquest conjunt de dades ha sigut generat artificialment a partir de soroll Gaussià del detector de Virgo real sumat a models aproximats de *glitches* (5.1). Aquests *glitches* han sigut modelats per aproximar els sorolls transitoris més comuns en l'espai freqüència-temps. També s'ha inclòs una categoria de *Chirps* per representar la coalescència d'un sistema binari compacte.

¹https://figshare.com/articles/dataset/Images_of_simulated_glitches_in_Numpy_format/7166210

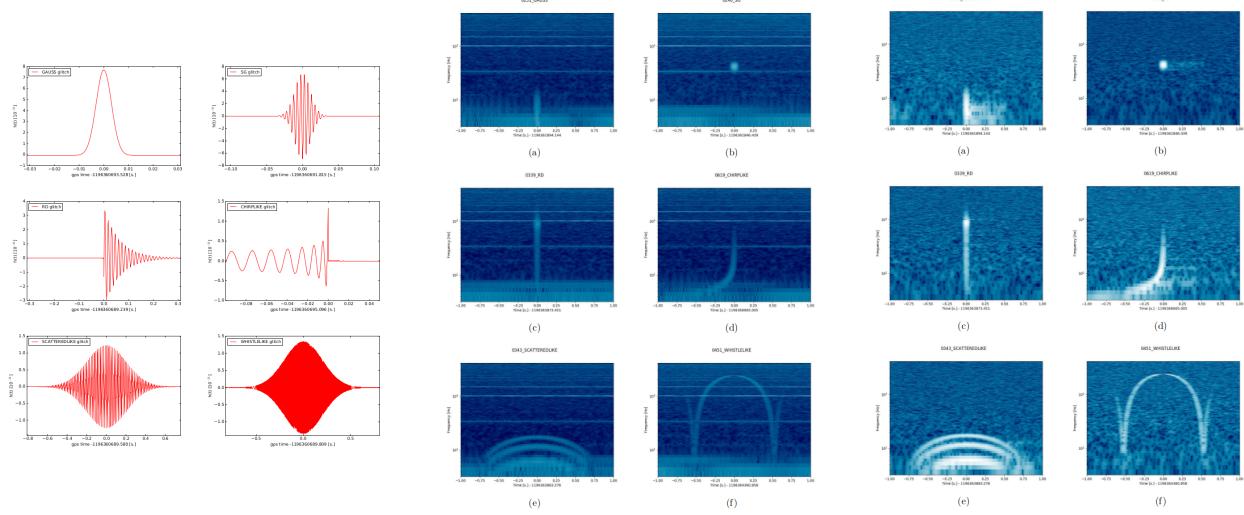


Figura 5.1: Exemples de les 6 famílies de *glitches* generades al conjunt de dades en [35].

A l'esquerra es mostra la sèrie temporal corresponent a un *glitch* de cada família, al centre es mostra l'espectrogràma generat a partir de sumar aquestes sèries temporals al soroll del detector generat i a la dreta els espectrogrames processats com s'indica en 5.1.1

La tasca a realitzar és la classificació d'imatges d'espectrogrames en 6 famílies de *glitches* (incloent-hi el chirp) i una classe addicional només amb soroll.

Per crear el conjunt de dades en primer lloc es genera un soroll Gaussià amb freqüència de mostreig 8192Hz amb una *ASD* corresponent a la ([sensibilitat del detector LIGO H1](#)), que es pot aconseguir aplicant un filtre FIR (filtre d'impuls finit) adient a soroll blanc. Un filtre és una operació aplicada sobre un senyal que dona lloc a un senyal nou, que pot ser utilitzat per modificar el contingut de freqüències. Un filtre FIR té és un tipus de filtre que quan rep un senyal d'impuls, dona lloc a un senyal amb un nombre finit de valors no nuls. Aquests filters poden ser dissenyats i aplicats digitalment de forma que el senyal resultant tinga el *ASD* desitjat (per més detalls, vegeu [9] Ch. 2 i Ch. 10).

A continuació s'afegeixen *glitches* en instants en el temps generats per una distribució de Poisson amb freqüència mitjana 0.5Hz de forma que puga haver *glitches* propers en el temps. La tria de cada tipus de *glitch* es fa de forma aleatòria de forma que totes les classes es representen per igual, triant 2000 imatges de cadascuna amb paràmetres aleatoris dins d'un rang adient.

Per aconseguir una imatge de cada tipus de *glitch*, seleccionem un interval de 2 segons al voltant de cadascun. Per augmentar la robustesa dels models entrenats, aquestes imatges han estat lleugerament desplaçades en un 10 % de la seua mida. Afegim també imatges de soroll que no conté *glitches*, de forma que obtenim un conjunt de dades de 14000 imatges. De cara a l'entrenament aquest conjunt s'ha dividit en proporcions 70 : 15 : 15 per entrenar, avaluar i testejar respectivament.

Així, cada imatge es tracta d'una matriu $x \in \mathbb{R}^{241 \times 531}$ d'amplaria 241 i altura 531. Diferents tècniques de tractament de dades són necessàries per poder preparar els espectrogrames:

1. Blanquejar la sèrie temporal: les dades contenen gran quantitat de soroll no transitori especialment en les baixes freqüències. Blanquejar-lo en el domini temporal com descrit en (3.3) millorarà la qualitat de les dades
2. Calcular l'espectrogràma de Fourier: els *glitches* són distingibles a l'espai de freqüències, en aquest cas

triem calcular l'espectrograma basat en el *PSD* (3.6) de la sèrie temporal en diversos intervals.

3. Normalitzar valors: podem aplicar una normalització a les dades per millorar el contrast entre els *glitches* i el soroll de fons. Per fer-ho, primer transformem els valors dels píxels a una escala de grisos amb 32 bits, i a continuació dividim els valors dels píxels que es troben entre el percentil 5 i 95 per la seu mediana.
4. Reduir dimensions de les imatges: per eficiència, reduïm les dimensions de les imatges a 132×282 píxels.

En aquest cas utilitzarem el conjunt de dades proporcionat per [35] on els dos primers passos del tractament de dades ja han estat aplicats. Així, ens limitem a aplicar els últims dos passos en la nostra implementació.

5.1.2. Gravity Spy

Gravity Spy és un projecte de caracterització de *glitches* utilitzant les dades de LIGO Livingston i Hanford en que es classifiquen les diferents classes de *glitches* tant amb voluntaris com amb models de *machine learning*.

Per trobar potencials *glitches*, s'utilitza l'algorisme Omicron [39] que detecta intervals de temps de les dades amb un excés de potència. A continuació aquests intervals poden ser estudiats a partir dels seus espectrogrames, anomenats Omega scans els quals es gasten per caracteritzar els *glitches*. Amb l'addició de dades dels diversos períodes de detecció es poden entrenar models cada volta més potents i crear un catàleg de *glitches* encara més gran. D'aquesta forma Gravity Spy recopila i prepara conjunts de dades d'alta qualitat per entrenar models de classificació, en particular utilitzarem un conjunt recopilat a Octubre del 2018 [30] i estudiat en [29] (utilitzem aquest conjunt de dades en lloc de la versió actualitzada [31] degut a tenir menor grandària i donar pitjor resultats).

Aquest conjunt de dades conté 22 classes de *glitches* (5.2) i per a cada event disposem de l'espectrograma construït amb la Q-transformació (Secció 3.5) centrat en el *glitch* amb resolucions temporals $0,5s, 1s, 2s$ i $4s$. Cada espectrograma és de 140×170 píxels. En aquest cas les dades ja han sigut tractades de forma adient per al seu ús com a conjunt d'entrenament. Notem que la distribució de *glitches* per classes és molt desequilibrada (5.1), que caldrà compensar a l'hora d'entrenar un model amb l'ús de varies tècniques.

Glitch	Entrenament	Validació	Testeig
1080Lines	230	49	49
1400Ripples	162	35	35
Air Compressor	41	8	9
Blip	1308	281	280
Chirp	46	10	10
Extremely Loud	318	68	68
Helix	195	42	42
Koi Fish	581	125	124
Light Modulation	401	86	86
Low Frequency Burst	460	99	98
Low Frequency Lines	317	68	68
No Glitch	127	27	27
None of the Above	62	13	13
Paired Doves	19	4	4
Power Line	317	68	68
Repeating Blips	200	43	42
Scattered Light	321	69	69
Scratchy	248	53	53
Tomte	81	17	18
Violin Mode	330	71	71
Wandering Line	31	6	7
Whistle	213	46	46
Total	6008	1288	1287

Cuadro 5.1: Distribució de les classes de *glitches* en les particions de dades d'entrenament, validació i testeig.

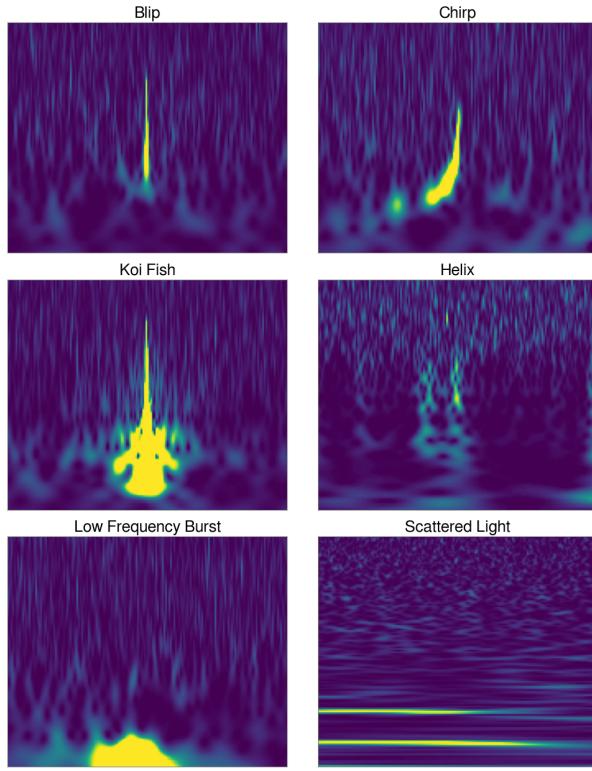


Figura 5.2: Exemples de 6 classes de *glitches* del conjunt de dades de Gravity Spy amb resolució temporal de 0,5s.

5.2. Mètrica d'avaluació

Donada una predicció de les classes d'un grup de dades, podem definir la seu matriu de confusió com una matriu C en que cada fila representa les classes correctes de les dades i cada columna les classes predites. Així l'element C_{ij} representa el nombre d'elements que pertanyen a la classe i i classificats en la classe j , de forma que un model perfecte donaria lloc a prediccions amb una matriu de confusió diagonal. Per visualitzar amb més facilitat es comú normalitzar aquesta matriu, en aquest treball normalitzarem les files de forma que els elements de cada fila sumen 1.

Si només tinguerem dues classes, negatiu (N) i positiu (P) aleshores la matriu tindria la següent forma

	Predicció positiva	Predicció negativa
Classe positiva	TP	FN
Classe negativa	FP	TN

Amb múltiples classes podem definir classes binàries definint les classes $P = \{x|x \text{ pertany a la classe } i\}$ i $N = \{x|x \text{ no pertany a la classe } i\}$ per a cada classe i . Així podem estudiar la precisió i altres estadístiques per

a cada classe considerant la corresponent classe binària. Una mètrica molt útil és el *F1-score*, definit per

$$F_1 = \frac{2}{\frac{1}{P} + \frac{1}{R}} = 2 \frac{P \cdot R}{P + R}$$

on

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN}.$$

Aquest combina informació de la precisió P , la proporció de classificacions correctes dels elements classificats com a positius i el reclam o sensibilitat R , la proporció de classificacions correctes dels elements positius. En aquest treball utilitzem principalment la *macro F-score* calculant la precisió i reclam mitjans (*macro-average*) e introduint aquests en la fórmula, és a dir

$$\bar{F}_1 = \frac{1}{m} \sum_{i=1}^m F_i = \frac{1}{m} \sum_{i=1}^m \frac{P_i \cdot R_i}{P_i + R_i}$$

on els subíndexos sobre TP, FP, FN representen els corresponents valors per a cada classe $i = 1, \dots, m$. Aquesta forma de calcular la *F1-score* front a calcular la mitjana de les precisions i reclams sobre les classes sol tenir un millor comportament en conjunts de dades amb distribucions de classes molt desequilibrades [42].

5.3. Desequilibri de classes

Com que al conjunt de dades de Gravity Spy trobem una gran asimetria al nombre de *glitches* de cada classe, el model assignarà menys importància a aprendre les classes més infreqüents. Per compensar aquest problema, existeixen diverses estratègies que podem aplicar.

D'una banda podem sobrerepresentar les classes poc freqüents durant l'entrenament. Una forma fàcil de fer-ho és assignar una probabilitat a cada classe inversament proporcional al nombre de glitches que hi ha en la classe a la que pertany. Si denotem $n_i = |\{x | x \text{ pertany a la classe } i\}|$, definim la probabilitat \bar{p}_k de triar una classe k amb

$$p_k = \frac{1}{n_k}, \quad \bar{p}_k = \frac{p_k}{\sum_{i=1}^m p_i}.$$

Aleshores durant l'entrenament, cada element es triat de forma uniforme dins de la classe k triada amb probabilitat \bar{p}_k .

Per altra banda podem implementar una funció de cost que penalitza amb major pes l'error sobre les classes poc freqüents, forçant al model a no ignorar-les.

$$J_\eta(y, p) = - \sum_{k=1}^m w_k y_k \log(p_k)$$

on w_k regulen la intensitat de la penalització a cada classe. En aquest cas utilitzem la ponderació proposada en [36], donada per

$$w_k = \frac{1 - \eta}{1 - \eta^{n_k}}, \quad \bar{w}_k = \frac{w_k}{\sum_{i=1}^m w_i}$$

on el paràmetre η regula l'efecte de la ponderació i triem $\eta = 0,99$.

5.4. Funció de cost modificada

Tot i que les imatges de *glitches* seleccionades en Gravity Spy han sigut seleccionades per estar classificades amb alta confiança, es possible que continguen errors, o que en alguns casos les classes s'assemblien molt. En aquests casos el model es penalitzat d'igual forma que si haguera triat qualsevol altra classe, i l'entrenament pot ser dificultat. Per admetre aquests casos i no ser penalitzats tan fortament, podem suavitzar la distribució objectiu i exigir una menor confiança en les prediccions. Per aplicar aquesta tècnica, anomenada *label smoothing* apliquem la següent transformació a la distribució de probabilitats correcta, de forma que si la distribució correcta d'una dada que pertany a la classe $1 \leq j \leq n$, aleshores la distribució correcta passa a ser

$$y_\alpha = \left(\frac{\alpha}{n-1}, \dots, \frac{\alpha}{n-1}, 1-\alpha, \frac{\alpha}{n-1}, \dots, \frac{\alpha}{n-1} \right).$$

Un altre enfoc que podem aplicar és modificar la funció de cost per penalitzar les classes que resulten més fàcils de classificar. Podem modificar la funció de cost de forma que les prediccions amb alta confiança resulten en un cost major, que pot resultar molt útil en un cas com aquest en que hi ha morfologies de *glitches* molt més diferenciables que altres. Aquest tipus de funció de cost s'anomena *focal loss* i va ser introduïda en [33] per solucionar el desequilibri entre la classe “fons de l'imatge” i “imatge d'interés” en tasques de reconeixement d'imatges. Aquesta funció de cost es defineix com

$$J_\gamma(y, p) = - \sum_{k=1}^m (1-p_i)^\gamma y_k \log(p_k)$$

on γ regula la penalització a les classificacions amb alta confiança.

Notem que podem combinar aquest enfoc amb la funció de cost ponderada i obtindre

$$J_{\gamma, \eta}(y, p) = - \sum_{k=1}^m w_k (1-p_i)^\gamma y_k \log(p_k).$$

5.5. Augmentació d'imatges

La grandària d'un conjunt de dades és un factor fonamental per poder entrenar un bon model sobre aquest, quant més gran siga millor podrà aprendre la distribució d'aquest. Una forma d'augmentar la grandària d'un conjunt d'imatges és aplicar transformacions a aquestes, si la distribució de les dades té propietats d'invariància respecte d'aquestes, es possible aconseguir inclús millor resultats [14].

Proposem les següents transformacions (vegeu un exemple en Figura 5.3):

- Transformació de color: un canvi aleatori sobre la lluminositat, contrast, saturació i to de l'imatge.
- Rotació: una rotació aleatoria de ± 5 graus.
- Retall: un retall d'un factor d'entre 0,7 i 1 de la mida original.

Aquestes només seran aplicades durant l'entrenament i el seu ús serà un altre hiperparàmetre a optimitzar. Notem també quan utilitzem els models que combinen punts de vista sempre apliquem aquestes transformacions individualment a cada imatge i no a la seua combinació.

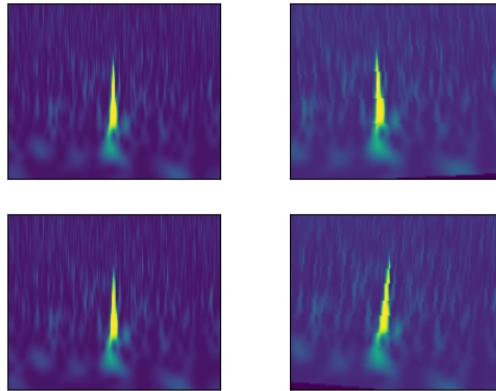


Figura 5.3: Exemples de transformacions per augmentar imatges aplicades al conjunt de dades de Gravity Spy. Dalt a l'esquerra es mostra un *glitch* de tipus *Blip*, les altres imatges són exemples de les transformacions aleatòries descrites en Secció 5.5.

5.6. Models tipus *VGG*

Per considerar l'arquitectura més simple d'una xarxa convolucional proposem una sèrie de models similars al model *VGG* (Figura 4.7), és a dir, basats en blocs de capes convolucionals.

D'una banda considerem una xarxa convolucional amb tres blocs que anomenem $3 \times \text{CNN}$, basat en el proposat en [35] per classificar el conjunt de dades de *glitches* sintètics. L'arquitectura d'aquest model és de tres blocs convolucionals i un bloc final tipus *MLP* (5.4), en què els kernels de les capes convolucionals són de grandària 3×3 . Per regularitzar el model s'inclouen capes de *Dropout* al final de cada bloc convolucional i després de la capa amagada del bloc de *MLP*.

De cara a la classificació del conjunt de dades de Gravity Spy, proposem un model similar entrenat sobre els espectrogrames dels *glitches* d'una sola resolució. L'arquitectura d'aquest model és més simple (5.5) i l'anomenem $2 \times \text{CNN}$ 1-canal.

5.7. Fusió de múltiples punts de vista

Els *glitches* presenten morfologies de diverses duracions en el temps i que potser són més distingibles si els visualitzem des de diferents punts de vista, que disposem al conjunt de Gravity Spy. Peraprofitar diversos punts de vista en un model *CNN* els podem incorporar al model de diverses formes, segons en quina profunditat del model combinem la informació d'aquests. Aquest enfoc es explorat sobre el conjunt de dades Gravity Spy en [27] i els models proposats en aquesta secció es basen en les arquitectures proposades en aquest article.

Es poden combinar els diversos punts de vista al principi combinant-les en una imatge més gran e introduint aquesta “superimatge” al model, que anomenem *CNN-fusió* inicial (Figura 5.6). Per crear la imatge més gran podem simplement juntar les imatges de les diferents resolucions apegant-les dos a dos dues voltes formant una imatge de dimensions 280×340 píxels i utilitzar aquesta com a valor d'entrada al primer bloc convolucional. Per estructurar el model utilitzem una sèrie de blocs convolucionals, on cadascun té una capa convolucional (k, n_c, p), seguida d'una capa de normalització *MaxPool* amb kernel 2×2 . Després dels blocs, com és usual, vectorizem els resultats i els introduïm en dues capes totalment connectades amb 256 neurones cadascuna que utilitzen *dropout*.

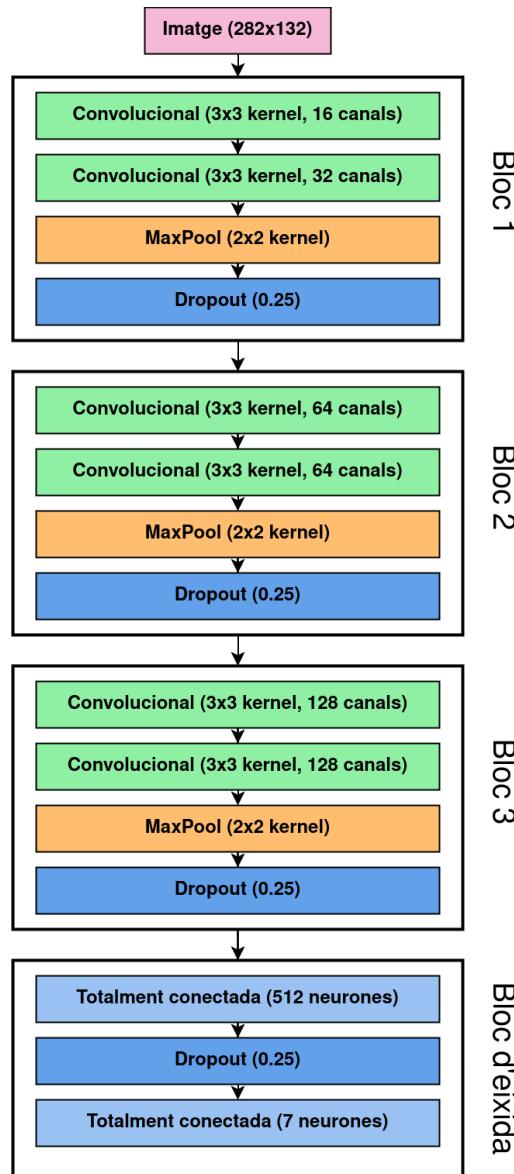


Figura 5.4: Arquitectura del model 3×CNN per classificar els *glitches* sintètics.

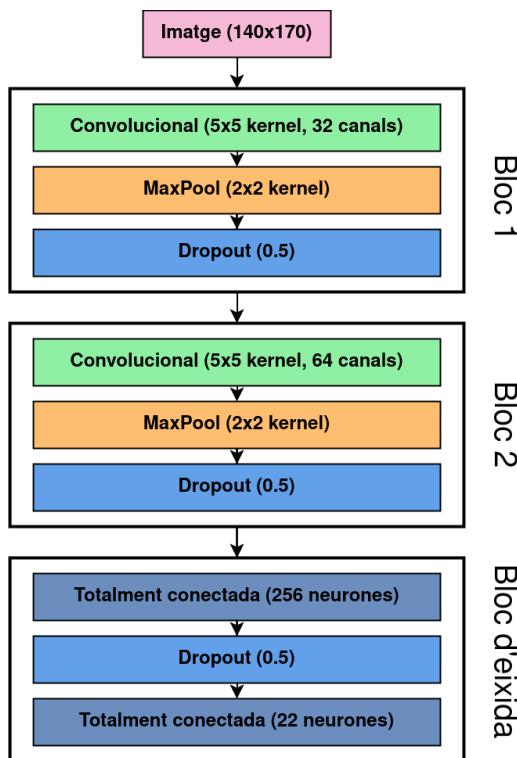


Figura 5.5: Model tipus $2 \times \text{CNN}$ 1-canal utilitzat per classificar espectrogrames de Gravity Spy d'una resolució.

També podem entrenar un conjunt de blocs convolucionals per a cada punt de vista i unir-los per obtindre una representació combinada interna al model, de la qual seguir aplicant més capes del tipus desitjat. Així podem crear una sèrie de blocs convolucionals que extrauen informació de cada punt de vista i unir el resultat de cadascun en la dimensió dels canals, de forma que si cada cap dona lloc a una representació interna amb dimensions $n_{\text{latent}} \times W \times H$ els podem combinar en un tensor de dimensions $(4 \cdot n_{\text{latent}}) \times W \times H$ el qual es gasta com a valor d'entrada en el següent bloc. A aquest model l'anomenem CNN-fusió intermitja (Figura 5.7). Per als blocs convolucionals de cada cap i post-fusió l'estructura és similar a la descrita en la fusió inicial, igualment per a la secció *MLP* final.

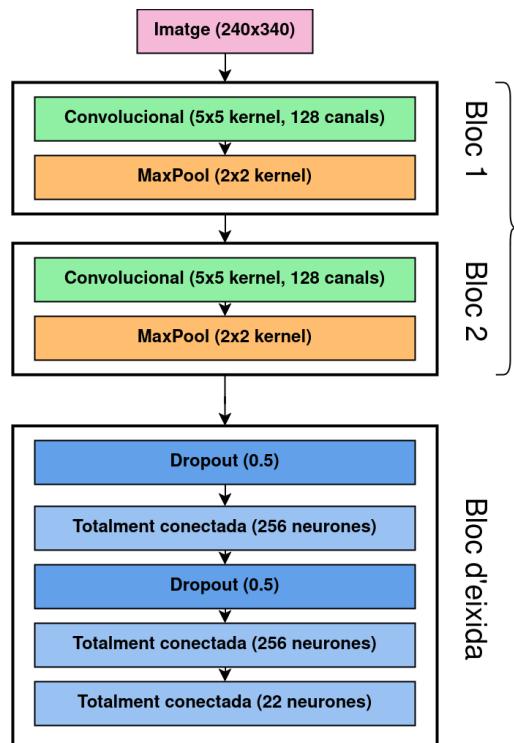


Figura 5.6: Arquitectura del model CNN-fusió inicial tipus (5, 128), (5, 128).

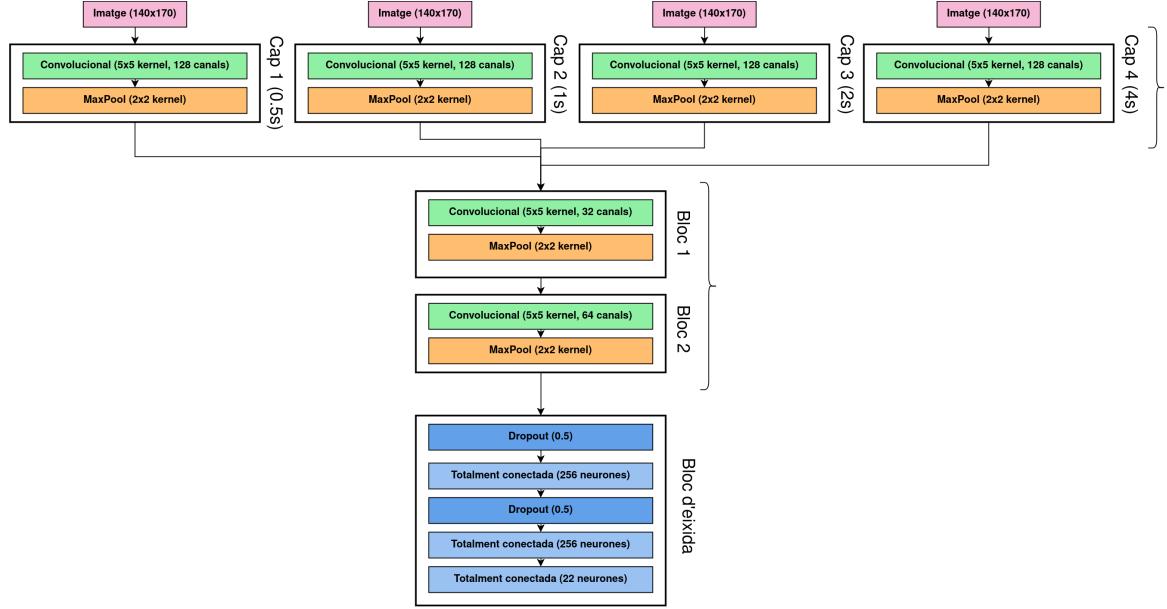


Figura 5.7: Arquitectura del model CNN-fusió intermitja tipus (5, 128) en cada cap i (5, 32), (4, 64) en la part post-fusió.

5.8. Mecanisme d'atenció

Arrel de l'enfoc de multiples punts de vista, en [46] va ser proposada l'idea d'incloure un mecanisme d'atenció entrenable, de forma que el model puga aprendre quins punts de vista diferèncien més clarament cada *glitch*.

Podem millorar l'efectivitat de l'ús de múltiples punts de vista afegint una capa al model que actue com a un mecanisme d'atenció sobre cada punt de vista. La idea és la següent: si en lloc de un valor $x \in \mathbb{R}^D$ associat a una classe binària $y \in \{0, 1\}$, diposem d'una col·lecció de mostres $X = \{x_1, \dots, x_K\}$ independents i que no tenen ordre, amb classes associades $y_i \in \{0, 1\}$ cadascuna, aleshores definim la etiqueta de la col·lecció per

$$Y = \begin{cases} 0 & \text{si } \sum_i y_i = 0 \\ 1 & \text{sinó} \end{cases} = \max_i(y_i)$$

Ara, la funció màxim no és diferenciable i dona problemes si la utilitzem com a part d'una funció objectiu, doncs entrenem un model per modelar $\theta(X)$ suposant que Y segueix una distribució de Bernoulli amb aquest paràmetre. Notem que $\theta(X) \in \mathbb{R}$ és una funció invariant respecte a permutacions en X i en aquest cas el següent teorema ens pot guiar per construir un model adient.

Teorema 5.8.1. *Siga una funció $S(X) \in \mathbb{R}$ sobre $X = \{x_1, \dots, x_k\}$. Aleshores és invariant respecte a permutacions si, i només si existeixen funcions g, f adients tal que*

$$S(X) = g \left(\sum_{i=1}^K f(x_i) \right)$$

Podem substituir aquestes funcions f i g per xarxes neuronals i entrenar-les sobre el conjunt de dades. Per implementar aquest model extraurem de cada instància x_i una representació interna z_i amb blocs convolucionals

i vectoritzant el resultat d'aquests (colapsant totes les dimensions per obtindre un vector de dimensions el producte de les originals). Doncs ara definim el mecanisme d'atenció com en [32]:

$$\mathcal{M}_{Att} = \sum_{i=1}^4 \alpha_i z_i,$$

on els coeficients α_i modelen la importància que li dona el model a cada punt de vista i són també entrenats per paràmetres $w \in \mathbb{R}^L, V \in \mathbb{R}^{L \times D}, U \in \mathbb{R}^{L \times D}$

$$\alpha_i = \frac{\exp(w^T (\tanh(Vz_i^T)) \odot \sigma(Uz_i^T))}{\sum_{j=1}^4 \exp(w^T (\tanh(Vz_j^T)) \odot \sigma(Uz_j^T))}$$

on \odot és el producte component a component i L ens permet controlar la dimensió interna al mecanisme d'atenció. Aquesta tria bé de substituir la funció f per una *MLP* amb una capa amagada d'una neurona i aplicant la funció *softmax* als coeficients de forma que sumen 1 i siguin independents del nombre d'instàncies. Ara com que la funció d'activació *tanh* és quasi lineal a l'interval $[0, 1]$ multipliquem el resultat d'aquesta neurona per el d'una amb funció d'activació sigmoidal.

Incloent aquest component com a una capa que combina les quatre branques corresponents a cada resolució, podem construir un model amb l'estructura mostrada en Figura 5.8, que anomenem CNN atenció.

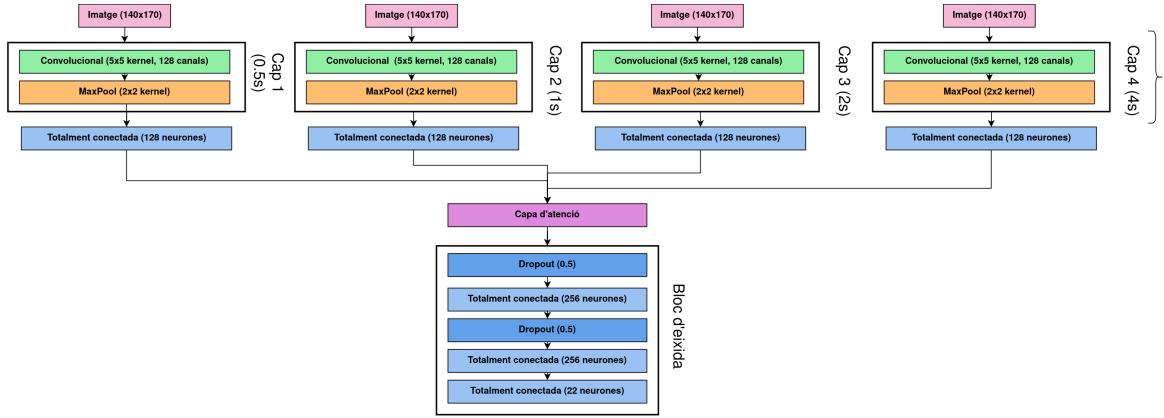


Figura 5.8: Model CNN atenció amb arquitectura (5, 128) per classificar el conjunt de dades de Gravity Spy.

5.9. Transferència d'aprenentatge

Una estratègia molt eficient per entrenar xarxes neuronals aplicades a imatges és prendre un model preentrenat en un conjunt de dades d'imatges generiques molt gran i refinar-ho al conjunt de dades més reduït d'interès. Per fer-ho, normalment es “congelen” totes les capes del model excepte l’última capa totalment connectada, no modificant els seus paràmetres durant la retropropagació. Després, si el model dona bons resultats, es pot entrenar la xarxa sencera amb un ratio d'aprenentatge més reduït.

En aquest cas hem provat aquest enfoc amb els models ResNet18 (Figura 4.10), ResNet50 i vit_b_16 [41].

La majoria d'aquests models estan entrenats amb imatges amb tres canals (*RGB*), per adaptar-los a imatges d'un canal tenim dues opcions. La primera opció és copiar les imatges als tres canals i utilitzar la capa inicial

original. Alternativament, podem redefinir la primera capa per admetres només un canal i entrenar-la des de zero tal i com fem amb l'última. Podem veure l'arquitectura del model ResNet18 adaptat d'aquestes dues formes en la figura 5.9.

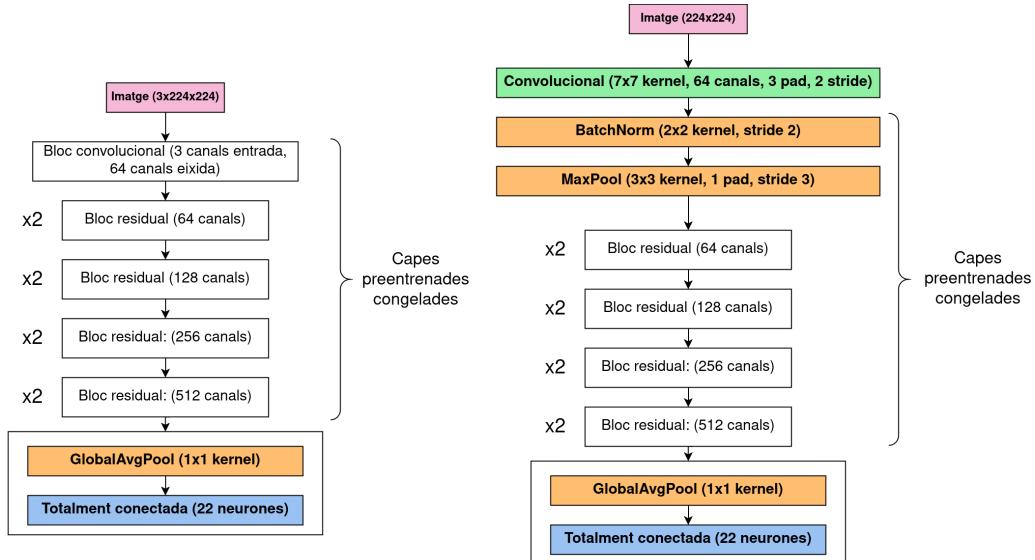


Figura 5.9: Arquitectura del model ResNet18 preentrenat en la configuració d'entrenament inicial. A l'esquerra es mostra el model que rep 3 canals d'entrada i a la dreta l'adaptació a 1 canal d'entrada.

Capítol 6

Resultats

Tots els models d'aquesta secció han sigut entrenats utilitzant ordinadors virtuals amb dues unitats de processament gràfic (GPU T4x2) disponibles en *jupyter notebooks* que proporciona la plataforma [Kaggle](#). Per poder replicar els resultats de forma local, tot el codi és disponible al repositori [Gravitational-Waves-Glitch-Classification](#). Notem també que hem utilitzat precisió mixta [34] per reduir el temps d'entrenament dels models.

6.1. Classificació de *glitches sintètics*

El conjunt de dades d'espectrogrames de *glitches* sintètics resulta molt més fàcil de classificar que el de Gravity Spy per contindre relativament poques classes i per ser generades en base a models conegeuts. Com que en aquest cas les classes es troben molt equilibrades, podem utilitzar la precisió com a mètrica d'avaluació de forma intercanviable amb la *F1 – score*.

El model 3xCNN (Fig. 5.4) dona millors resultats que el model 1xCNN. Els millors hiperparàmetres trobats per a aquest model són $lr = 0,0008$, $weight\ decay = 0$ i $batch\ size = 128$. La funció de cost *focal loss* i el *label smoothing* no han millorat els resultats, pel que hem utilitzat la funció de cost per defecte. L'optimitzador Adam ha donat els millors resultats de convergència i hem vist que 20 èpoques han resultat suficients per aconseguir la millor precisió possible. Triem el model utilitzant *early stopping*, és a dir, triant el model de l'època d'entrenament en que s'assoleix la màxima precisió d'avaluació.

Amb aquesta configuració, s'aconsegueix una precisió de 99,9 % sobre el conjunt de testeig (6.1). Podem observar també en (6.2) que l'error que comet aquest model és només que una confusió amb alguns cassos de soroll tipus Gauss amb tipus RD.

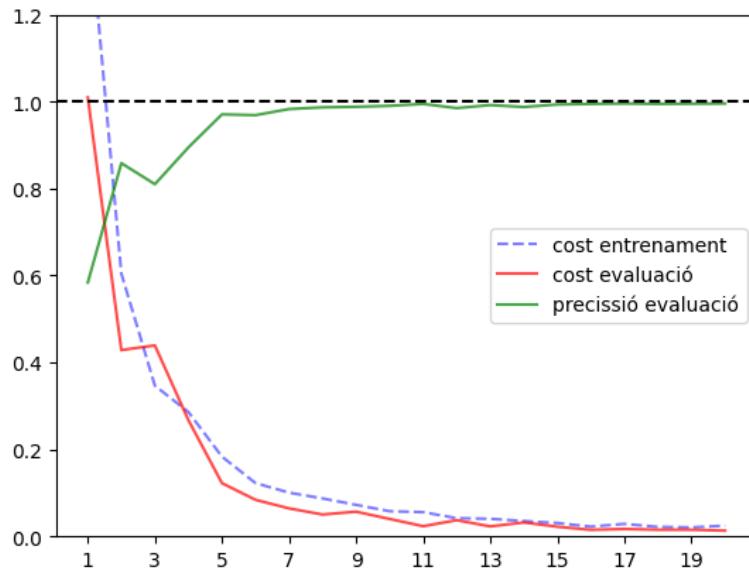


Figura 6.1: Funció de cost i precisió durant l'entrenament del model 3xCNN.

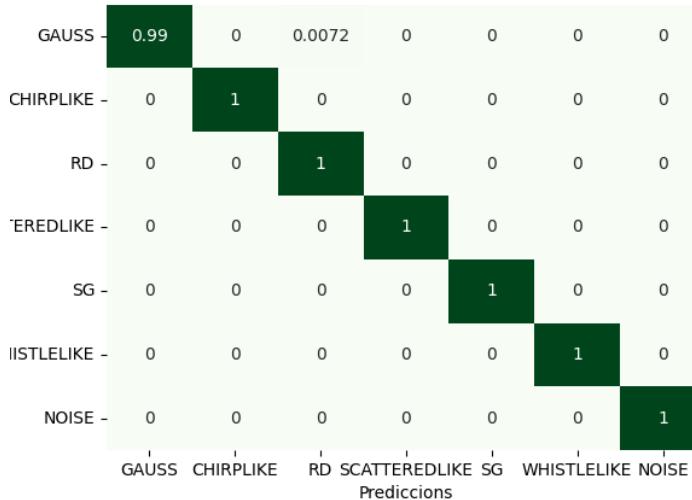


Figura 6.2: Matriu de confussió del model 3xCNN sobre les dades de testeig.

Per avaluar la variabilitat de l'error del model, també entrenem 10 models amb la mateixa configuració i calculem la precisió mitjana i la seua desviació típica. Per comparar amb l'efectivitat d'altres models provem a entrenar un model similar però només amb un bloc convolucional que anomenem 1xCNN i provem amb un altre model clàssic de classificació, el model Random Forest [8] implementat amb la llibreria scikit-learn [18] de python en la figura (6.3).

	Precisió	Precisió (10 models)
3xCNN	99,90 %	$99,42 \pm 0,64 \%$
1xCNN	97,3 %	-
RF	99,44 %	-

Figura 6.3: Precisió de diversos models entrenats sobre el conjunt de dades de *glitches* sintètics.

6.2. Classificació de Gravity Spy

La configuració que hem triat per defecte per als models d'aquesta secció és $lr = 0,001$, $weight\ decay = 2 \cdot 10^{-2}$, $batch\ size = 64$, $label\ smoothing = 0$, mostreig de paràmetres ponderat per classe, l'optimitzador $AdamW$, la funció de cost *cross-entropy loss* ponderada per classe, $dropout = 0,25$, l'ús augmentació d'imatges, 50 èpoques d'entrenament i una reducció del 50 % de lr programada si l'error d'entrenament no es redueix durant dues èpoques.

En primer lloc hem entrenat models tipus 2xCNN 1-canal utilitzant només que una resolució i aplicant augmentació d'imatges, ja que ha donat resultats superiors al processament per defecte. En aquest cas la funció de cost no s'ha ponderat per classe per dificultar l'entrenament dels models. Podem veure els resultats obtinguts per a cada resolució disponible en (6.4). La resolució que permet la millor classificació sembla ser la de 1 segon, açò pot ser degut a una falta de context en la resolució 0,5s (veure Figura 6.5) i una proporció excessiva de fons a les resolucions superiors de 2s i 4s (veure Figura 6.6).

Mostrem també els models *ensemble* de totes les resolucions i sense la resolució de 4s, és a dir prenen la prediccio de màxima confiança d'entre les prediccions de cada model triat. Aquests models donen millors resultats, indicants que cal combinar l'informació dels diferents punts de vista per aconseguir millors resultats, com farem amb les següents estratègies.

Model	Precisió	F_1 -score
Resolució 0,5s	95,18 %	93,75 %
Resolució 1s	96,74 %	94,50 %
Resolució 2s	95,96 %	93,78 %
Resolució 4s	94,48 %	90,49 %
<i>Ensemble</i> 0,5s+1s+2s+4s	96,74 %	94,38 %
<i>Ensemble</i> 0,5s+1s+2s	96,81 %	95,31 %

Figura 6.4: Taula de resultats dels models 2xCNN 1-canal segons la resolució i mode utilitzats al conjunt de dades Gravity Spy.

A continuació mostrem les configuracions que han donat millors resultats als models de fusió de punts de vista i atenció:

- Fusió inicial: $dropout = 0,5$, sense augmentació d'imatges i amb una arquitectura tipus $(5, 32, 2), (5, 64, 2), (5, 128, 2), (5, 256, 2), (5, 256)$.
- Fusió intermitja: una arquitectura tipus $(5, 64, 2), (5, 128, 2)$ en cada cap i $(5, 128, 2), (5, 128), (5, 256)$ en la secció post-fusió.
- Atenció: una arquitectura $(3, 64), (5, 128), (7, 64)$ als caps, $lr = 0,003$, 128 neurones a l'espai latent d'atenció i 128 neurones a la component amagada d'atenció.

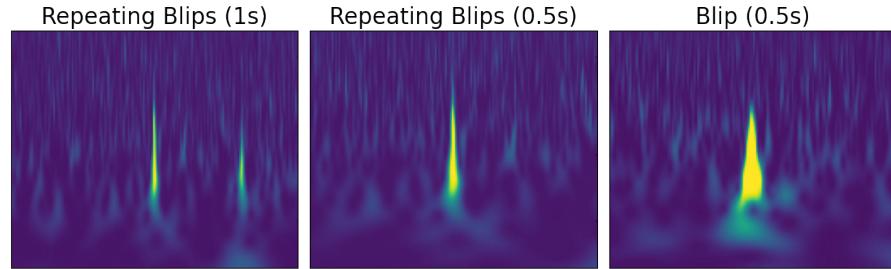


Figura 6.5: Exemple de les limitacions d'utilitzar una única resolució massa menuda per classificar un *glitch*, a l'esquerra podem veure una imatge d'un *Repeating Blip* i observem que utilitzant la resolució de 0,5s (imatge central) resulta indistingible d'un *Blip* (imatge a la dreta).

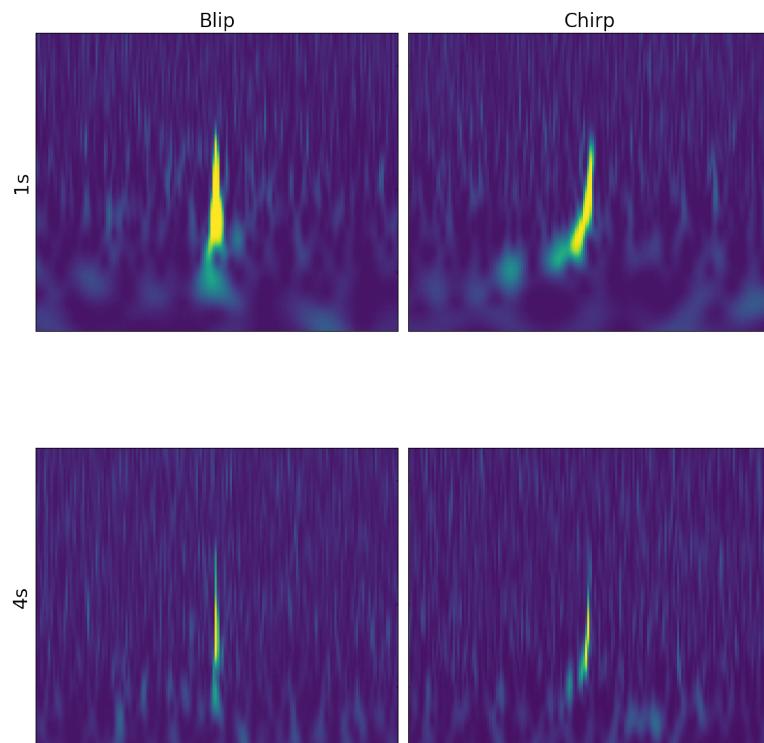


Figura 6.6: Exemple de les limitacions d'utilitzar una única resolució massa gran per classificar un *glitch*, la diferència entre les classes *Chirp* i *Blip* és molt més clara en la resolució d'1 que en la resolució de 4 segons.

- Atenció VGG: una arquitectura $3 \times (3, 64), 3 \times (3, 64), 3 \times (3, 256), 3 \times (3, 256)$ VGG als caps, *label smoothing*= 0,1, *dropout*= 0,3 i 256 neurones a l'espai latent d'atenció i 128 neurones a la component amagada d'atenció.

Resulta també interessant mostrar el funcionament del mecanisme d'atenció. Mostrem els pesos d'atenció mitjans per a cada vista per classe durant el testeig al model atenció VGG en Figura 6.8. Podem observar que efectivament dona més importància a les vistes més relevantes per a certs *glitches*, per exemple en la classe *Paired Doves*, ja que sovint només és distingible en la resolució més gran (veure Figura 6.9).

Podem veure els resultats d'aquests models en la Taula 6.7 on s'aprecia que el millor model és CNN-fusió intermitja.

Model	Precisió	F_1 -score
CNN-fusió intermitja	97,44 %	96,52 %
CNN-fusió inicial	97,59 %	96,37 %
CNN-atenció	95,37 %	91,02 %
CNN-atenció VGG	96,97 %	95,33 %

Figura 6.7: Taula de resultats dels models de fusió de punts de vista i atenció sobre Gravity Spy.

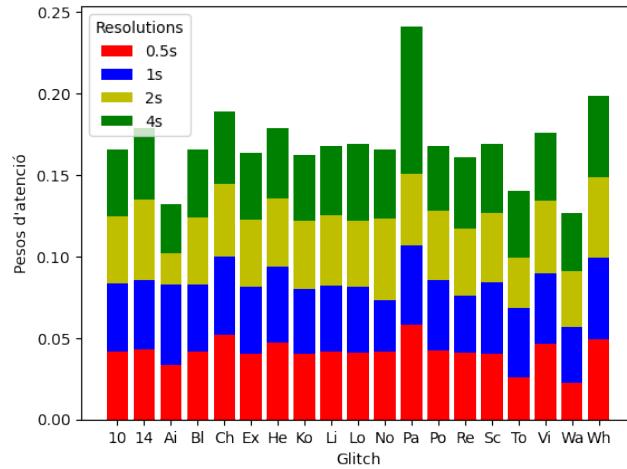


Figura 6.8: Pesos d'atenció mitjans segons la classe de *glitch* durant el testeig del model CNN-atenció VGG.

Finalment entrenem els models més moderns. Per a aquests hem decidit utilitzar l'adaptació de les dades a 3 canals triplicant les imatges i utilitzant fusió inicial a les imatges. El millor model ResNet18 utilitza augmentació d'imatges, funció de cost ponderada per classe i $lr = 0,0001$.

Per aplicar transferència d'aprenentatge sobre models tipus ResNet18, ResNet50 i vit_b_16. Per fer-ho, s'han entrenat els models congelats durant 40 èpoques amb $lr = 10^{-2}$ quan s'ha observar convergència i després 40 èpoques més entrenant el model sencer però amb $lr = 10^{-4}$. En aquests casos la funció de cost *focal loss* ha donat millors resultats. Podem veure que el model ResNet18 dona els millors resultats en Taula 6.10, amb la matriu de confusió mostrada en Figura 6.11.

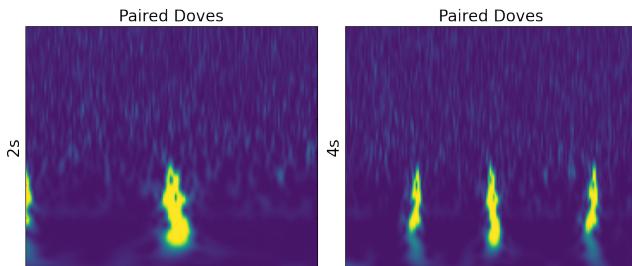


Figura 6.9: Exemple d'un *glitch* tipus *Paired Doves* a diferents resolucions, on s'aprecia que el punt de vista més important correspon a la resolució de 4s.

Model	Precisió	F_1 -score
ResNet18	98,06 %	97,59 %
ResNet18 (preentrenat)	98,06 %	97,68 %
ResNet50 (preentrenat)	97,70 %	96,01 %
vit_b_16 (preentrenat)	97,31 %	95,80 %

Figura 6.10: Taula de resultats dels models residuals i *ViT* sobre Gravity Spy.

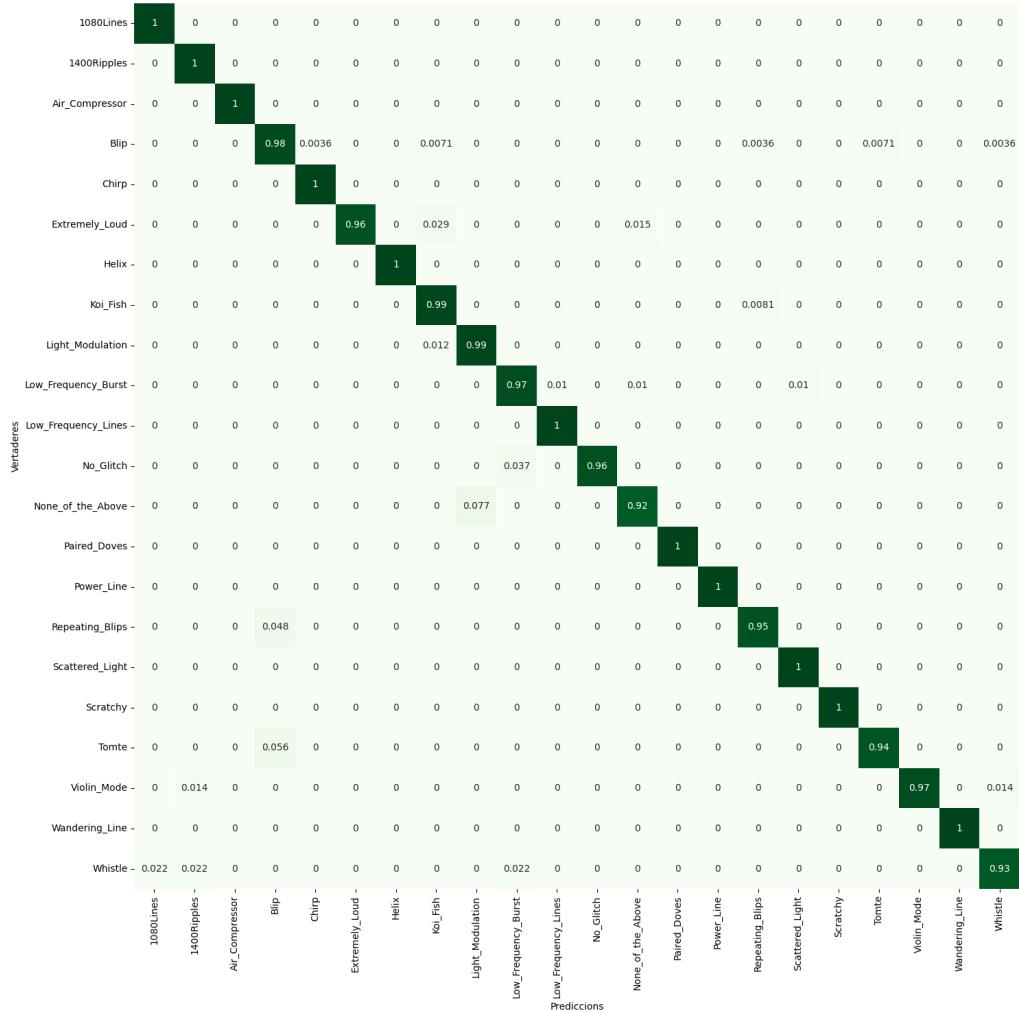


Figura 6.11: Matriu de confusió del millor model ResNet18 sobre les dades de testeig de Gravity Spy.

Capítol 7

Conclusions

La classificació dels *glitches* en dades d'ones gravitacionals continuará sent una tasca d'interés amb la construcció de detectors més precisos i la caracterització de més morfologies de *glitches*. Hem recopilat els millors models CNN utilitzats en la literatura científica, que aniran ampliant-se i millorant-se amb noves estructures CNN, estratègies innovadores i avanços al *machine learning*. Tot i que les xarxes neuronals són caixes negres i és molt difícil extraure conclusions d'aquests, podem destacar que l'ús de mecanismes d'atenció pot permetre determinar el rang de temps relevant per classificar cada *glitch*. També cal explorar amb més profunditat l'ús de *Vision Transformers* amb arquitectures més adients per aquesta tasca, que en aquest treball no hem pogut realitzar i poden resultar interessants per generar un espai de codificació dels *glitches* de dimensió reduïda. Aquest últim punt també ha estat limitat per l'ús exclusiu de recursos de computació gratuïts al núvol. Amb accés a *GPs* més potents es podrien haber explorat més models i arribat a arquitectures més òptimes per a aquells triats.

En aquest treball només que hem utilitzat dades gravitacionals a l'espai de freqüències i hem realitzat classificació d'imatges sobre aquest espai. Aquest enfoc sembla poc eficient tenint en compte que l'origen de les dades és una sèrie temporal de gran resolució temporal. Gran part de la informació és perduda i resumida en el contingut de freqüències, a partir del qual els models infereixen diferents estadístics relevantes. Es podria triar un enfoc en que es gastaren directament les sèries temporals o es combinaren amb les seues freqüències, com a dades d'entrada d'un model. Resulta interessant com en altres tasques, com l'eliminació del soroll produït per un *glitch* capturat també en un canal auxiliar aquest enfoc és possible, treballant sobre l'espai temporal i optimitzant el model utilitzant el seu contingut de freqüències per construir una funció de cost [40], [38].

Aquest enfoc requereix l'ús de canals auxiliars per entrenar models de classificació. Aquests canals ja són utilitzats per vetar dades amb alt soroll a partir de la seua correlació amb el canal principal, pel que seria interessant crear models capaços d'aprendre els diferents tipus d'acoplaments de sorolls entre canals. Cal remarcar que aquests canals no són públics en la majoria de casos excepte excepcionalment per realitzar recerca, i cal arribar a un acord amb l'organització corresponent per fer ús d'ells durant un període de temps estableert.

En qualsevol cas, amb el pas del temps i els esforços de la comunitat científica per recopilar dades d'alta qualitat, conjunts de dades més grans i útils seran disponibles. Aquestes dades donaran lloc a la possibilitat d'entrenar millors models, i aquests seran basats en aquells que funcionen als conjunts de dades actuals.

Aquest treball en definitiva es centra en un petit component de la detecció d'ones gravitacionals, i l'objectiu últim seria una classificació completa dels *glitches* conegeuts, que permetria l'eliminació d'aquests amb tractament de dades i milloraria el funcionament dels vetos per determinar la qualitat de les dades. És així un àrea que permetrà i espentàrà l'avanç de la resta d'àrees necessàries per detectar events gravitacionals amb confiança i precisió.

Bibliografia

- [1] Albert Einstein. «The Field Equations of Gravitation». En: *Sitzungsber. Preuss. Akad. Wiss. Berlin (Math. Phys.)* 1915 (1915), págs. 844-847.
- [2] C.E. Shannon. «Communication in the Presence of Noise». En: *Proceedings of the IRE* 37.1 (ene. de 1949). Conference Name: Proceedings of the IRE, págs. 10-21. ISSN: 2162-6634. DOI: [10.1109/JRPROC.1949.232969](https://doi.org/10.1109/JRPROC.1949.232969). URL: <https://ieeexplore.ieee.org/document/1697831> (visitado 11-06-2024).
- [3] G. Cybenko. «Approximation by superpositions of a sigmoidal function». En: *Mathematics of Control, Signals, and Systems* 2.4 (dic. de 1989), págs. 303-314. ISSN: 0932-4194, 1435-568X. DOI: [10.1007/BF02551274](https://doi.org/10.1007/BF02551274). URL: <http://link.springer.com/10.1007/BF02551274> (visitado 01-04-2024).
- [4] Kouichi Yamaguchi et al. «A neural network for speaker-independent isolated word recognition». En: *First International Conference on Spoken Language Processing (ICSLP 1990)*. First International Conference on Spoken Language Processing (ICSLP 1990). ISCA, 18 de nov. de 1990, págs. 1077-1080. DOI: [10.21437/ICSLP.1990-282](https://doi.org/10.21437/ICSLP.1990-282). URL: https://www.isca-archive.org/icslp_1990/yamaguchi90c_iclsp.html (visitado 18-06-2024).
- [5] Wei Zhang et al. «Parallel distributed processing model with local space-invariant interconnections and its optical architecture». En: *Applied Optics* 29.32 (10 de nov. de 1990), pág. 4790. ISSN: 0003-6935, 1539-4522. DOI: [10.1364/AO.29.004790](https://doi.org/10.1364/AO.29.004790). URL: <https://opg.optica.org/abstract.cfm?URI=ao-29-32-4790> (visitado 15-03-2024).
- [6] Peter J. Brockwell y Richard A. Davis. *Time Series: Theory and Methods*. Springer Series in Statistics. New York, NY: Springer New York, 1991. ISBN: 978-1-4419-0319-8 978-1-4419-0320-4. DOI: [10.1007/978-1-4419-0320-4](https://doi.org/10.1007/978-1-4419-0320-4). URL: <http://link.springer.com/10.1007/978-1-4419-0320-4> (visitado 12-06-2024).
- [7] Judith C. Brown. «Calculation of a constant Q spectral transform». En: *The Journal of the Acoustical Society of America* 89.1 (1 de ene. de 1991), págs. 425-434. ISSN: 0001-4966, 1520-8524. DOI: [10.1121/1.400476](https://doi.org/10.1121/1.400476). URL: <https://pubs.aip.org/jasa/article/89/1/425/678890/Calculation-of-a-constant-Q-spectral> (visitado 13-02-2024).
- [8] Leo Breiman. «Bagging predictors». En: *Machine Learning* 24.2 (ago. de 1996), págs. 123-140. ISSN: 0885-6125, 1573-0565. DOI: [10.1007/BF00058655](https://doi.org/10.1007/BF00058655). URL: <http://link.springer.com/10.1007/BF00058655> (visitado 14-05-2024).
- [9] John G. Proakis. *Digital Signal Processing: Principles, Algorithms, and Applications*. Google-Books-ID: sAcfAQAAIAAJ. Prentice Hall, 1996. 1040 págs. ISBN: 978-0-13-394289-7.
- [10] D.H. Wolpert y W.G. Macready. «No free lunch theorems for optimization». En: *IEEE Transactions on Evolutionary Computation* 1.1 (abr. de 1997), págs. 67-82. ISSN: 1089778X. DOI: [10.1109/4235.585893](https://doi.org/10.1109/4235.585893). URL: <https://ieeexplore.ieee.org/document/585893/> (visitado 15-03-2024).
- [11] Yann Lecun et al. «Gradient-Based Learning Applied to Document Recognition». En: *Proceedings of the IEEE* 86 (1 de dic. de 1998), págs. 2278-2324. DOI: [10.1109/5.726791](https://doi.org/10.1109/5.726791).
- [12] Maximilian Riesenhuber y Tomaso Poggio. «Hierarchical models of object recognition in cortex». En: *Nature Neuroscience* 2.11 (1 de nov. de 1999), págs. 1019-1025. ISSN: 1097-6256, 1546-1726. DOI: [10.1038/14819](https://doi.org/10.1038/14819). URL: https://www.nature.com/articles/nn1199_1019 (visitado 12-06-2024).

- [13] C. E. Shannon. «A mathematical theory of communication». En: *ACM SIGMOBILE Mobile Computing and Communications Review* 5.1 (ene. de 2001), págs. 3-55. ISSN: 1559-1662, 1931-1222. DOI: [10.1145/584091.584093](https://doi.org/10.1145/584091.584093). URL: <https://dl.acm.org/doi/10.1145/584091.584093> (visitado 15-03-2024).
- [14] P.Y. Simard, D. Steinkraus y J.C. Platt. «Best practices for convolutional neural networks applied to visual document analysis». En: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings*. Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings. Ago. de 2003, págs. 958-963. DOI: [10.1109/ICDAR.2003.1227801](https://doi.org/10.1109/ICDAR.2003.1227801). URL: <https://ieeexplore.ieee.org/document/1227801> (visitado 27-06-2024).
- [15] S Chatterji et al. «Multiresolution techniques for the detection of gravitational-wave bursts». En: *Classical and Quantum Gravity* 21.20 (21 de oct. de 2004), S1809-S1818. ISSN: 0264-9381, 1361-6382. DOI: [10.1088/0264-9381/21/20/024](https://doi.org/10.1088/0264-9381/21/20/024). URL: <https://iopscience.iop.org/article/10.1088/0264-9381/21/20/024> (visitado 23-02-2024).
- [16] *Pattern Recognition and Machine Learning*. Springer New York, 2006. DOI: [10.1007/978-0-387-45528-0](https://doi.org/10.1007/978-0-387-45528-0). URL: <https://link.springer.com/deleted> (visitado 04-05-2024).
- [17] Thomas W. Baumgarte y Stuart L. Shapiro. *Numerical Relativity: Solving Einstein's Equations on the Computer*. 1.^a ed. Cambridge University Press, 24 de jun. de 2010. ISBN: 978-0-521-51407-1 978-1-139-19334-4. DOI: [10.1017/CBO9781139193344](https://doi.org/10.1017/CBO9781139193344). URL: <https://www.cambridge.org/core/product/identifier/9781139193344/type/book> (visitado 11-06-2024).
- [18] Fabian Pedregosa et al. «Scikit-learn: Machine Learning in Python». En: *Journal of Machine Learning Research* 12.85 (2011), págs. 2825-2830. ISSN: 1533-7928. URL: <http://jmlr.org/papers/v12/pedregosa11a.html> (visitado 14-05-2024).
- [19] Olga Russakovsky et al. «Detecting Avocados to Zucchinis: What Have We Done, and Where Are We Going?». En: *2013 IEEE International Conference on Computer Vision*. 2013 IEEE International Conference on Computer Vision (ICCV). Sydney, Australia: IEEE, dic. de 2013, págs. 2064-2071. ISBN: 978-1-4799-2840-8. DOI: [10.1109/ICCV.2013.258](https://doi.org/10.1109/ICCV.2013.258). URL: <http://ieeexplore.ieee.org/document/6751367/> (visitado 18-06-2024).
- [20] Nitish Srivastava et al. «Dropout: a simple way to prevent neural networks from overfitting». En: *Journal of machine learning research* (2014). URL: <https://www.semanticscholar.org/paper/Dropout%3A-a-simple-way-to-prevent-neural-networks-Srivastava-Hinton/34f25a8704614163c4095b3ee2fc969b60de4698> (visitado 01-05-2024).
- [21] Sergey Ioffe y Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2 de mar. de 2015. arXiv: [1502.03167\[cs\]](https://arxiv.org/abs/1502.03167). URL: <http://arxiv.org/abs/1502.03167> (visitado 19-06-2024).
- [22] Karen Simonyan y Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 10 de abr. de 2015. arXiv: [1409.1556\[cs\]](https://arxiv.org/abs/1409.1556). URL: <http://arxiv.org/abs/1409.1556> (visitado 18-06-2024).
- [23] J. Veitch et al. «Parameter estimation for compact binaries with ground-based gravitational-wave observations using the LALInference software library». En: *Physical Review D* 91.4 (6 de feb. de 2015), pág. 042003. ISSN: 1550-7998, 1550-2368. DOI: [10.1103/PhysRevD.91.042003](https://doi.org/10.1103/PhysRevD.91.042003). URL: <https://link.aps.org/doi/10.1103/PhysRevD.91.042003> (visitado 11-06-2024).
- [24] The LIGO Scientific Collaboration y the Virgo Collaboration. «Observation of Gravitational Waves from a Binary Black Hole Merger». En: *Physical Review Letters* 116.6 (11 de feb. de 2016), pág. 061102. ISSN: 0031-9007, 1079-7114. DOI: [10.1103/PhysRevLett.116.061102](https://doi.org/10.1103/PhysRevLett.116.061102). arXiv: [1602.03837\[astro-ph,physics:gr-qc\]](https://arxiv.org/abs/1602.03837). URL: [http://arxiv.org/abs/1602.03837](https://arxiv.org/abs/1602.03837) (visitado 01-03-2024).
- [25] Kaiming He et al. «Deep Residual Learning for Image Recognition». En: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). ISSN: 1063-6919. Jun. de 2016, págs. 770-778. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90). URL: <https://ieeexplore.ieee.org/document/7780459> (visitado 19-06-2024).

- [26] Peter Shawhan. «On the Hunt for Counterparts to Gravitational-Wave Events». LUVOIR Seminar. U. of Maryland / Joint Space-Science Institute, 7 de dic. de 2016. URL: https://asd.gsfc.nasa.gov/luvoir/events/seminars/2016/Shawhan_LUVOIR_Dec2016.pdf.
- [27] Sara Bahaadini et al. *Deep Multi-view Models for Glitch Classification*. 28 de abr. de 2017. arXiv: [1705.00034\[cs\]](https://arxiv.org/abs/1705.00034). URL: <http://arxiv.org/abs/1705.00034> (visitado 07-04-2024).
- [28] Alex Krizhevsky, Ilya Sutskever y Geoffrey E. Hinton. «ImageNet classification with deep convolutional neural networks». En: *Communications of the ACM* 60.6 (24 de mayo de 2017), págs. 84-90. ISSN: 0001-0782, 1557-7317. DOI: [10.1145/3065386](https://doi.org/10.1145/3065386). URL: <https://dl.acm.org/doi/10.1145/3065386> (visitado 18-06-2024).
- [29] S. Bahaadini et al. «Machine learning for Gravity Spy: Glitch classification and dataset». En: *Information Sciences* 444 (mayo de 2018), págs. 172-186. ISSN: 00200255. DOI: [10.1016/j.ins.2018.02.068](https://doi.org/10.1016/j.ins.2018.02.068). URL: <https://linkinghub.elsevier.com/retrieve/pii/S0020025518301634> (visitado 07-04-2024).
- [30] Sara Bahaadini et al. *Machine Learning For Gravity Spy: Glitch Classification And Dataset*. 31 de oct. de 2018. DOI: [10.5281/ZENODO.1476156](https://doi.org/10.5281/ZENODO.1476156). URL: <https://zenodo.org/record/1476156> (visitado 26-06-2024).
- [31] Scott Coughlin. *Gravity Spy Training Set*. Ver. v1.1.0. 1 de nov. de 2018. DOI: [10.5281/ZENODO.1486046](https://doi.org/10.5281/ZENODO.1486046). URL: <https://zenodo.org/record/1486046> (visitado 07-04-2024).
- [32] Maximilian Ilse, Jakub M. Tomczak y Max Welling. *Attention-based Deep Multiple Instance Learning*. 28 de jun. de 2018. arXiv: [1802.04712\[cs,stat\]](https://arxiv.org/abs/1802.04712). URL: <http://arxiv.org/abs/1802.04712> (visitado 13-04-2024).
- [33] Tsung-Yi Lin et al. *Focal Loss for Dense Object Detection*. 7 de feb. de 2018. arXiv: [1708.02002\[cs\]](https://arxiv.org/abs/1708.02002). URL: <http://arxiv.org/abs/1708.02002> (visitado 16-06-2024).
- [34] Paulius Micikevicius et al. *Mixed Precision Training*. 15 de feb. de 2018. arXiv: [1710.03740\[cs,stat\]](https://arxiv.org/abs/1710.03740). URL: <http://arxiv.org/abs/1710.03740> (visitado 16-06-2024).
- [35] Massimiliano Razzano y Elena Cuoco. «Image-based deep learning for classification of noise transients in gravitational wave detectors». En: *Classical and Quantum Gravity* 35.9 (10 de mayo de 2018), pág. 095016. ISSN: 0264-9381, 1361-6382. DOI: [10.1088/1361-6382/aab793](https://doi.org/10.1088/1361-6382/aab793). arXiv: [1803.09933\[astro-ph,physics:gr-qc\]](https://arxiv.org/abs/1803.09933). URL: <http://arxiv.org/abs/1803.09933> (visitado 15-03-2024).
- [36] Yin Cui et al. «Class-Balanced Loss Based on Effective Number of Samples». En: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, CA, USA: IEEE, jun. de 2019, págs. 9260-9269. ISBN: 978-1-72813-293-8. DOI: [10.1109/CVPR.2019.00949](https://doi.org/10.1109/CVPR.2019.00949). URL: <https://ieeexplore.ieee.org/document/8953804/> (visitado 25-06-2024).
- [37] The LIGO Scientific Collaboration et al. «A guide to LIGO-Virgo detector noise and extraction of transient gravitational-wave signals». En: *Classical and Quantum Gravity* 37.5 (5 de mar. de 2020), pág. 055002. ISSN: 0264-9381, 1361-6382. DOI: [10.1088/1361-6382/ab685e](https://doi.org/10.1088/1361-6382/ab685e). arXiv: [1908.11170\[astro-ph,physics:gr-qc\]](https://arxiv.org/abs/1908.11170). URL: <http://arxiv.org/abs/1908.11170> (visitado 04-03-2024).
- [38] Rich Ormiston et al. «Noise Reduction in Gravitational-wave Data via Deep Learning». En: *Physical Review Research* 2.3 (14 de jul. de 2020), pág. 033066. ISSN: 2643-1564. DOI: [10.1103/PhysRevResearch.2.033066](https://doi.org/10.1103/PhysRevResearch.2.033066). arXiv: [2005.06534\[astro-ph,physics:gr-qc,physics:physics\]](https://arxiv.org/abs/2005.06534). URL: <http://arxiv.org/abs/2005.06534> (visitado 16-03-2024).
- [39] Florent Robinet et al. «Omicron: A tool to characterize transient noise in gravitational-wave detectors». En: *SoftwareX* 12 (jul. de 2020), pág. 100620. ISSN: 23527110. DOI: [10.1016/j.softx.2020.100620](https://doi.org/10.1016/j.softx.2020.100620). URL: <https://linkinghub.elsevier.com/retrieve/pii/S2352711020303332> (visitado 03-04-2024).
- [40] Gabriele Vajente et al. «Machine-learning non-stationary noise out of gravitational wave detectors». En: *Physical Review D* 101.4 (18 de feb. de 2020), pág. 042003. ISSN: 2470-0010, 2470-0029. DOI: [10.1103/PhysRevD.101.042003](https://doi.org/10.1103/PhysRevD.101.042003). arXiv: [1911.09083\[astro-ph,physics:gr-qc,physics:physics\]](https://arxiv.org/abs/1911.09083). URL: <http://arxiv.org/abs/1911.09083> (visitado 16-03-2024).
- [41] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 3 de jun. de 2021. arXiv: [2010.11929\[cs\]](https://arxiv.org/abs/2010.11929). URL: <http://arxiv.org/abs/2010.11929> (visitado 19-06-2024).

- [42] Juri Opitz y Sebastian Burst. *Macro F1 and Macro F1*. 8 de feb. de 2021. arXiv: [1911.03347\[cs,stat\]](https://arxiv.org/abs/1911.03347). URL: <http://arxiv.org/abs/1911.03347> (visitado 25-06-2024).
- [43] F. Acernese et al. *Virgo Detector Characterization and Data Quality: results from the O3 run*. 14 de oct. de 2022.
- [44] Maria Refinetti, Alessandro Ingrosso y Sebastian Goldt. *Neural networks trained with SGD learn distributions of increasing complexity*. 26 de mayo de 2023. arXiv: [2211.11567\[cond-mat,stat\]](https://arxiv.org/abs/2211.11567). URL: <http://arxiv.org/abs/2211.11567> (visitado 29-06-2024).
- [45] Aston Zhang et al. *Dive into Deep Learning*. Cambridge University Press, 2023.
- [46] Yunan Wu et al. *Advancing Glitch Classification in Gravity Spy: Multi-view Fusion with Attention-based Machine Learning for Advanced LIGO's Fourth Observing Run*. 23 de ene. de 2024. arXiv: [2401.12913\[astro-ph,physics:gr-qc\]](https://arxiv.org/abs/2401.12913). URL: <http://arxiv.org/abs/2401.12913> (visitado 19-03-2024).
- [47] Nils Andersson. *Gravitational-Wave Astronomy Exploring the Dark Side of the Universe*. Oxford University Press.