دانشگاه صنعتی امیرکبیر
(پلی‌تکنیک تهران)
دانشکده علوم کامپیوتر

پروژه کارشناسی
گرایش الگوریتم و نظریه محاسبه

رویکردی نوین بر امنیت اطلاعات: از پیچیدگی تجزیه اعداد تا یادگیری عمیق ماشین

پروژه درس نرم افزار ریاضی

نگارش
پرهام رحیمی

استاد راهنما
بهزاد نجفی سقزچی

استاد درس
امین حسینقلی زاده

خرداد ۱۴۰۵

بسم الله الرحمن الرحیم

فهرست مطالب

عنوان | صفحه

۱ **مقدمه و مفاهیم اولیه** ۱
۱-۱ تاریخچه و اهمیت تجزیه اعداد ۲
۱-۲ تعاریف پایه ریاضی ۳
۱-۳ ساختار مستند ۴

۲ **مدل‌سازی ریاضی و الگوریتم‌ها** ۵
۲-۱ مقدمه و مدل‌سازی ریاضی ۶
۲-۱-۱ توابع چندضابطه‌ای و حد ۶
۲-۱-۲ بسط چندجمله‌ای و انتگرال‌گیری ۶
۲-۱-۳ امید ریاضی و واریانس ۷
۲-۱-۴ تجزیه یک عبارت ۷
۲-۲ الگوریتم بهینه‌سازی و روندنما ۸
۲-۲-۱ شبه‌کد الگوریتم ۸
۲-۲-۲ روندنمای فرآیند ۱۰

۳ **توابع زیان و ارزیابی مدل‌ها** ۱۱
۳-۱ مقدمه ۱۲
۳-۲ منظم‌سازی و رگرسیون ریج ۱۲
۳-۳ تابع زیان هیوبر ۱۳
۳-۴ معادله نرمال ۱۴
۳-۵ الگوریتم‌های بهینه‌سازی ۱۴
۳-۶ معیارهای ارزیابی مدل‌ها ۱۵
۳-۶-۱ معیارهای ارزیابی در مسائل رگرسیون ۱۵
۳-۶-۲ معیارهای ارزیابی در مسائل دسته‌بندی ۱۶

۴ **شبکه‌های عصبی عمیق** ۱۷
۴-۱ مقدمه ۱۸
۴-۲ یاخته عصبی مصنوعی ۱۸
۴-۳ شبکه‌های عصبی چندلایه ۲۰
۴-۳-۱ ساختار لایه‌ها ۲۰
۴-۳-۲ توابع فعال‌ساز رایج ۲۰
۴-۳-۳ توابع زیان ۲۲
۴-۴ آموزش شبکه عصبی ۲۲

۴-۴-۱ انتشار رو به جلو و پس‌انتشار خطا ۲۲
۴-۴-۲ الگوریتم‌های بهینه‌سازی ۲۳
۴-۵ کاربردهای شبکه‌های عصبی عمیق ۲۴

۵ جمع‌بندی و نتیجه‌گیری و پیشنهادات ۲۵
۵-۱ جمع‌بندی و نتیجه‌گیری ۲۶
۵-۲ پیشنهادها برای کارهای آینده ۲۷

منابع و مراجع ۲۸

فهرست تصاویر

شکل صفحه

۱-۱ نمایش هندسی و الگوهای توزیع اعداد اول در مارپیچ اولام ۳
۱-۲ گراف ارتباط فصول و ساختار مستند ۴
۱-۲ فلوچارت فرآیند بهینه‌سازی و بررسی شرط توقف ۱۰
۱-۳ مقایسه رفتار توابع خطای مربعات ، خطای مطلق و زیان هیوبر ($\delta = ۱$). ۱۳
۱-۴ ساختار یاخته عصبی طبیعی در سیستم عصبی و مدل‌سازی یاخته عصبی مصنوعی . ۱۹

فهرست جداول

جدول صفحه

۳-۱ معرفی توابع زیان و روابط تحلیلی در ارزیابی مدل ۱۲
۳-۲ مقایسه الگوریتم‌های بهینه‌سازی مختلف ۱۵

فهرست نمادها

نماد	مفهوم
σ	تابع فعال‌ساز سیگموید
$\tanh$	تابع فعال‌ساز تانژانت هیپربولیک
α	نرخ یادگیری در الگوریتم‌های بهینه‌سازی
$\beta_۱, \beta_۲$	پارامترهای میانگین متحرک (نمایی) در الگوریتم بهینه‌ساز آدام
ϵ	مقدار ثابت کوچک برای جلوگیری از خطای تقسیم بر صفر در الگوریتم آدام
$\mathcal{L}$	تابع زیان مانند تابع زیان هیوبر
$\mathbb{R}$	مجموعه اعداد حقیقی

فصل اول

مقدمه و مفاهیم اولیه

در این فصل به بررسی مفاهیم پایه‌ای و تاریخچه تجزیه اعداد صحیح می‌پردازیم.

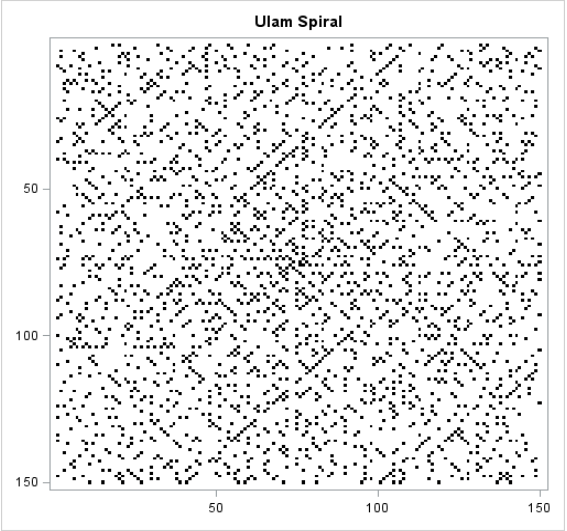
۱-۱ تاریخچه و اهمیت تجزیه اعداد

اعداد اول همواره به عنوان آجرهای سازنده ریاضیات و پایه و اساس نظریه اعداد شناخته می‌شوند. بر اساس اصول بنیادین ریاضی، هر عدد صحیحِ بزرگتر از یک، یا خود یک عدد اول است و یا می‌توان آن را به صورت ضرب چند عدد اول نوشت. فرایند یافتن این عوامل سازنده، «تجزیه عدد صحیح» [۱] نامیده می‌شود. در حالی که ضرب کردن دو عدد اولِ بسیار بزرگ به راحتی و در کسری از ثانیه توسط رایانه‌ها انجام می‌شود، فرایند معکوس آن، یعنی یافتن عوامل اولِ یک عدد بسیار بزرگ، مسئله‌ای به شدت زمان‌بر و از نظر محاسباتی پیچیده است. همین عدم تقارن محاسباتی، پایه و اساس سامانه های رمزنگاری کلید عمومی نوین را تشکیل می‌دهد که امروزه امنیت ارتباطات دیجیتال، تراکنش‌های بانکی و تبادل داده‌ها در اینترنت به آن وابسته است.[۵] با وجود اینکه اعداد اول نقش محوری در امنیت و محاسبات دارند، نحوه توزیع آن‌ها در میان اعداد طبیعی همواره یکی از اسرارآمیزترین مباحث ریاضیات بوده است. اعداد اول در نگاه اول رفتاری کاملاً تصادفی و بی‌نظم از خود نشان می‌دهند، اما ریاضی‌دانان با استفاده از نمایش‌های هندسی، الگوهای پنهان شگفت‌انگیزی را در میان آن‌ها کشف کرده‌اند. یکی از معروف‌ترینِ این نمایش‌ها «مارپیچ اولام» [۲] است که نشان می‌دهد توزیع اعداد اول کاملاً هم تصادفی نیست.[۸]

همان‌طور که در شکل ۱-۱ مشاهده می‌شود، اگر اعداد طبیعی را به صورت مارپیچی حول یک نقطه مرکزی بنویسیم و اعداد اول را علامت‌گذاری کنیم، الگوهای بصری خاصی نمایان می‌شود. در زیرشکل ۱-۱ (آ) ، نحوه ساختاردهی اولیه این مارپیچ و جایگاه اعداد اول (خانه‌های قرمزرنگ) را در یک مقیاس کوچک نشان می‌دهد. با گسترش این شبکه مارپیچی در مقیاس‌های بسیار بزرگتر، همان‌طور که در زیرشکل ۱-۱ (ب) قابل مشاهده است، اعداد اول به شکل معناداری روی خطوط قطری خاصی متراکم می‌شوند. این رفتار هندسی نشان می‌دهد که مسئله تجزیه اعداد و درک رفتار اعداد اول، فراتر از محاسبات پیچیده جبری، دارای ساختارهای عمیق هندسی و بصری نیز هست که الگوریتم‌های مدرن به دنبال کشف و بهره‌برداری از آن‌ها هستند.

[1] Integer Factorization

[2] Ulam Spiral



400	399	398	397	396	395	394	393	392	391	390	389	388	387	386	385	384	383	382	381
325	324	323	322	321	320	319	318	317	316	315	314	313	312	311	310	309	308	307	380
326	257	256	255	254	253	252	251	250	249	248	247	246	245	244	243	242	241	306	379
327	258	197	196	195	194	193	192	191	190	189	188	187	186	185	184	183	240	305	378
328	259	198	145	144	143	142	141	140	139	138	137	136	135	134	133	182	239	304	377
329	260	199	146	101	100	99	98	97	96	95	94	93	92	91	132	181	238	303	376
330	261	200	147	102	65	64	63	62	61	60	59	58	57	90	131	180	237	302	375
331	262	201	148	103	66	37	36	35	34	33	32	31	56	89	130	179	236	301	374
332	263	202	149	104	67	38	17	16	15	14	13	30	55	88	129	178	235	300	373
333	264	203	150	105	68	39	18	5	4	3	12	29	54	87	128	177	234	299	372
334	265	204	151	106	69	40	19	6	1	2	11	28	53	86	127	176	233	298	371
335	266	205	152	107	70	41	20	7	8	9	10	27	52	85	126	175	232	297	370
336	267	206	153	108	71	42	21	22	23	24	25	26	51	84	125	174	231	296	369
337	268	207	154	109	72	43	44	45	46	47	48	49	50	83	124	173	230	295	368
338	269	208	155	110	73	74	75	76	77	78	79	80	81	82	123	172	229	294	367
339	270	209	156	111	112	113	114	115	116	117	118	119	120	121	122	171	228	293	366
340	271	210	157	158	159	160	161	162	163	164	165	166	167	168	169	170	227	292	365
341	272	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	291	364
342	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	363
343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362

زیرشکل (آ) ساختار اولیه مارپیچ و جایگاه اعداد اول زیرشکل (ب) ظهور الگوهای قطری و تراکم اعداد اول (خانه‌های قرمزرنگ) در مقیاس کوچک در مقیاس بزرگتر

شکل ۱-۱: نمایش هندسی و الگوهای توزیع اعداد اول در مارپیچ اولام

۱-۲ تعاریف پایه ریاضی

برای درک بهتر الگوریتم‌های تجزیه، ابتدا باید برخی از مفاهیم اولیه نظریه اعداد را مرور کنیم. قضیه اساسی حساب بیان می‌کند که هر عدد صحیح و بزرگ‌تر از یک مانند n را می‌توان به صورت یکتایی (صرف‌نظر از ترتیب) به صورت حاصل‌ضرب اعداد اول نوشت.[۳] این موضوع در رابطه (۱-۱) نشان داده شده است:

$$n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k} = \prod_{i=1}^{k} p_i^{e_i} \tag{۱-۱}$$

که در آن نمادهای p_i اعداد اول متمایز و e_i توان‌های صحیح و مثبت هستند.

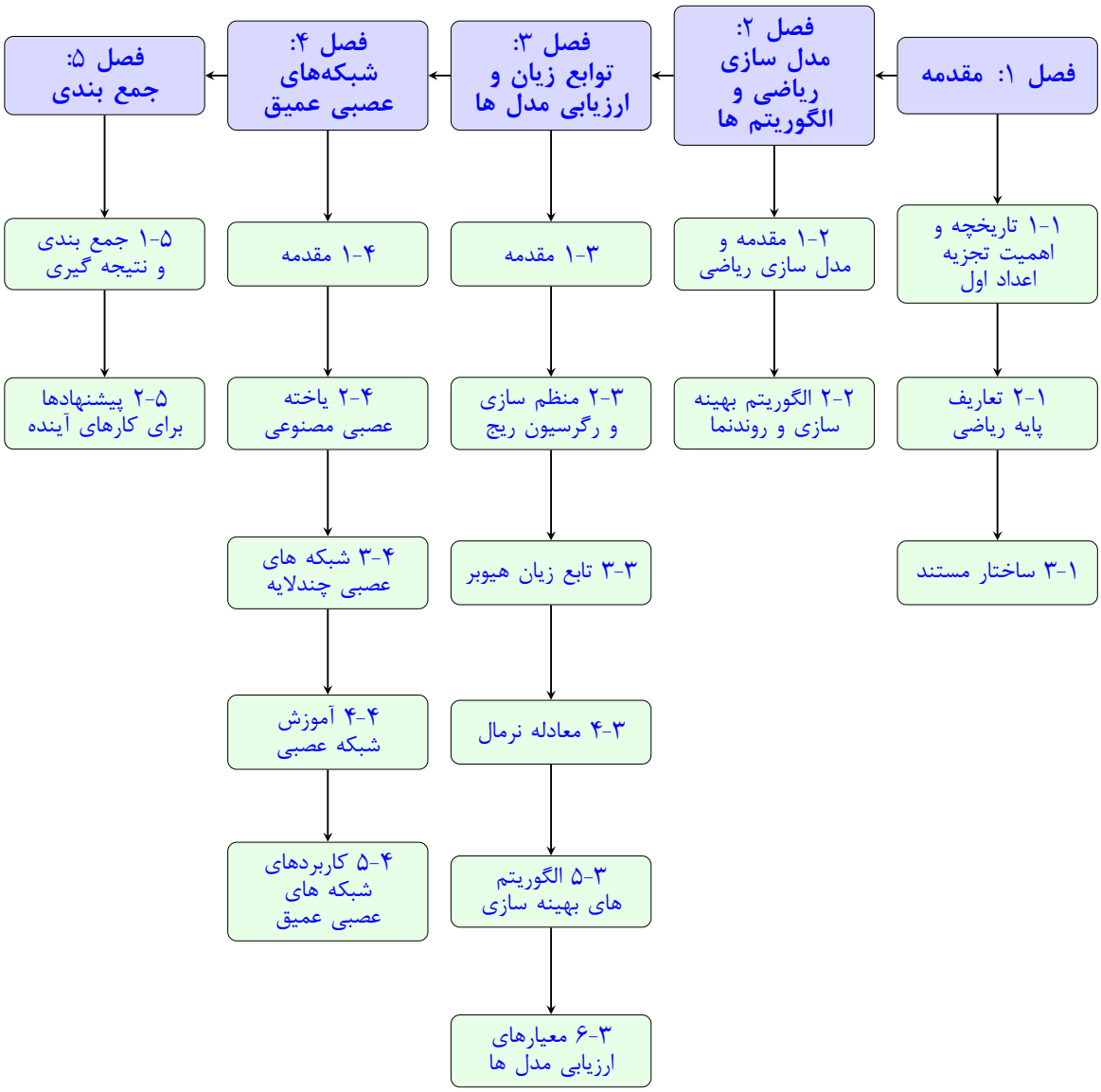
همچنین در بررسی الگوریتم‌ها، گاهی نیاز به تعریف توابع ریاضی برای جداسازی حالت‌های مختلف داریم. برای مثال، تابع نشانگر اول بودن یک عدد مانند $\mathrm{I}(n)$ را می‌توان به صورت یک تابع چندضابطه‌ای به شکل زیر تعریف کرد:

$$\mathrm{I}(n) = \begin{cases} 1 & \text{اگر } n \text{ یک عدد اول باشد} \\ 0 & \text{در غیر این صورت} \end{cases} \tag{۲-۱}$$

روابط (۱-۱) و (۱-۲) پایه و اساس درک الگوریتم‌هایی هستند که در فصل‌های بعدی به آن‌ها خواهیم پرداخت.

۱-۳ ساختار مستند

در رَوَندنمای[3] شکل ۱-۲ نمای کلی و سازمان‌دهی مطالب این مستند ارائه می‌شود تا خواننده دید روشنی نسبت به مسیر پیشبرد مباحث داشته باشد:



شکل ۱-۲: گراف ارتباط فصول و ساختار مستند

[3] Flowchart

فصل دوم

مدل‌سازی ریاضی و الگوریتم‌ها

۲-۱ مقدمه و مدل‌سازی ریاضی

در این فصل، به بررسی چارچوب ریاضی و مدل‌سازی‌های انجام‌شده در این مستند پرداخته می‌شود. هدف اصلی، ارائه یک ساختار تحلیلی منسجم برای توصیف رفتار سیستم و ارزیابی عملکرد آن است.

۲-۱-۱ توابع چندضابطه‌ای و حد

توابع چندضابطه‌ای در مدل‌سازی پدیده‌هایی که رفتار آن‌ها در بازه‌های مختلف تغییر می‌کند، کاربرد فراوانی دارند. فرض کنید x یک متغیر حالت باشد؛ در این صورت، تابع $f(x)$ را می‌توان به صورت رابطه (۲-۱) تعریف کرد:

$$f(x) = \begin{cases} x^{۲} + ۲x & x \geq ۰ \\ -x^{۳} & x < ۰ \end{cases}. \tag{۲-۱}$$

همان‌طور که مشاهده می‌شود، رفتار تابع در مرز $x = ۰$ تغییر می‌کند. برای بررسی رفتار مجانبی سامانه، محاسبه حد تابع در بی‌نهایت ضروری است که به صورت رابطه (۲-۲) بیان می‌شود:

$$\lim_{x \to \infty} f(x) = \infty. \tag{۲-۲}$$

۲-۱-۲ بسط چندجمله‌ای و انتگرال‌گیری

برای تقریب رفتار غیرخطی سامانه، می‌توان از بسط چندجمله‌ای استفاده نمود. فرض کنید N یک عدد صحیح و نشان‌دهنده مرتبه چندجمله‌ای باشد. در این حالت، چندجمله‌ای $P(x)$ به فرم رابطه (۲-۳) ارائه می‌گردد:

$$P(x) = \sum_{n=۰}^{N} a_n x^n. \tag{۲-۳}$$

به منظور محاسبه مساحت زیر نمودار یا ارزیابی انرژی سامانه در بازه $[a, b]$، از عملگر انتگرال‌گیری مطابق رابطه (۲-۴) استفاده می‌شود:

$$\int_a^b P(x)dx = F(b) - F(a). \tag{۲-۴}$$

۲-۱-۳ امید ریاضی و واریانس

برای تعمیم این مفاهیم و بررسی دقیق‌تر رفتارهای آماری، فرض کنید X یک متغیر تصادفی پیوسته با تابع چگالی احتمال[1] $f_X(x)$ باشد و $g(X)$ یک تابع پیوسته و دلخواه روی این متغیر تعریف نماید. بر اساس قضیه آماردان ناآگاه[2]، امید ریاضی برای تابع $g(X)$ با استفاده از انتگرال‌گیری بر روی کل دامنه متغیر تصادفی به شکل زیر محاسبه می‌گردد:

$$\mathbb{E}[g(X)] = \int_{-\infty}^{+\infty} g(x) f_X(x)\, dx, \tag{۲-۵}$$

همچنین با استفاده از گشتاور دوم، واریانس این تابع تعمیم‌یافته به صورت رابطه زیر به دست می‌آید:

$$\mathrm{Var}(g(X)) = \int_{-\infty}^{+\infty} \left(g(x) - \mathbb{E}[g(X)]\right)^{۲} f_X(x)\, dx. \tag{۲-۶}$$

همان‌طور که در روابط (۲-۵) و (۲-۶) مشاهده می‌شود، استفاده از ساختار انتگرالی به ما اجازه می‌دهد تا رفتار مدل را در فضاهای پیوسته با دقت بالاتری ارزیابی کنیم.

۲-۱-۴ تجزیه یک عبارت

در بسیاری از مسائل تحلیلی و محاسبات عددی، کار با یک فرم بسته و پیچیده ریاضی دشوار است. در این موارد، از تکنیک تجزیه کسر جزئی[3] برای شکستن یک عبارت گویای پیچیده به مجموعه‌ای از کسرهای ساده‌تر استفاده می‌شود. فرض کنید عبارت گویای $R(x)$ به صورت زیر تعریف شده باشد:

$$R(x) = \frac{x^{۲} + ۴x + ۱}{x^{۳} - x} \tag{۲-۷}$$

برای تجزیه این عبارت، ابتدا مخرج کسر را به عوامل اول تجزیه کرده و سپس کسر اولیه را به عنوان مجموعی از کسرهای ناشناخته با ضرایب مجهول بازنویسی می‌کنیم. این فرآیند بسط و تجزیه در چند

[1] Probability Density Function (PDF)

[2] Law of the Unconscious Statistician (LOTUS)

[3] Partial Fraction Decomposition

خط به صورت زیر قابل نمایش است:

$$\begin{aligned}\frac{x^{۲}+۴x+۱}{x^{۳}-x} &= \frac{x^{۲}+۴x+۱}{x(x-۱)(x+۱)}\\ &= \frac{A}{x}+\frac{B}{x-۱}+\frac{C}{x+۱}\\ &= \frac{A(x^{۲}-۱)+B(x^{۲}+x)+C(x^{۲}-x)}{x(x-۱)(x+۱)}\end{aligned} \tag{۲-۸}$$

همان‌طور که در رابطه (۲-۸) مشاهده می‌شود، با هم‌مخرج کردن سمت راست تساوی و برابر قرار دادن صورت کسرها، یک دستگاه معادلات خطی[۴] برای یافتن ضرایب A، B و C شکل می‌گیرد:

$$\begin{aligned} x^{۲}: &\quad A+B+C=۱,\\ x^{۱}: &\quad B-C=۴,\\ x^{۰}: &\quad -A=۱.\end{aligned}$$

حل دستگاه معادلات بالا به سادگی مقادیر مجهول را به دست می‌دهد و در نهایت عبارت پیچیده اولیه به فرمی ساده‌تر و قابل انتگرال‌گیری یا مشتق‌گیری تجزیه می‌گردد.

۲-۲ الگوریتم بهینه‌سازی و روندنما

برای یافتن مقادیر بهینه پارامترهای مدل، از الگوریتم‌های بهینه‌سازی[۵] مبتنی بر تکرار استفاده شده است. یکی از روش‌های کارآمد در این زمینه، روش گرادیان کاهشی[۶] است.[۶] در این بخش، شبه‌کد[۷] و روندنمای مربوط به این الگوریتم با در نظر گرفتن معیار توقف بر اساس خطای[۸] محاسباتی، ارائه می‌گردد.

۲-۲-۱ شبه‌کد الگوریتم

روند اجرایی روش گرادیان کاهشی در قالب الگوریتم ۱ تدوین شده است. در این ساختار، نماد e_{rr} نشان‌دهنده میزان خطای فعلی، ϵ آستانه تحمل خطا و $N_{\max}$ حداکثر تعداد تکرار مجاز است.

در الگوریتم ۱، شبه‌کد مربوط به روش بهینه‌سازی گرادیان کاهشی[۹] ارائه شده است. این روش، یک

[4] System of Linear Equations
[5] Optimization Algorithms
[6] Gradient Descent
[7] Pseudocode
[8] Error
[9] Gradient Descent

الگوریتم تکرارشونده برای یافتن نقطه کمینه محلی یک تابع هدفِ مشتق‌پذیر است.

همان‌طور که در روند اجرای این الگوریتم مشاهده می‌شود، کار با دریافت حدس اولیه برای پارامترها (θ_0)، نرخ یادگیری[10] (α) که طول گام‌ها را کنترل می‌کند، حد تحمل خطا (ϵ) و حداکثر تعداد تکرار مجاز ($N_{\max}$) آغاز می‌گردد. در ابتدا، مقدار خطای اولیه بی‌نهایت و شمارنده تکرارها صفر در نظر گرفته می‌شود.

در هر تکرار از حلقه، مراحل زیر اجرا می‌گردد:

ابتدا بردار گرادیان تابع هزینه J نسبت به پارامترها، یعنی $\nabla J(\theta_k)$، در نقطه فعلی محاسبه می‌شود. این بردار جهت بیشترین افزایش تابع را نشان می‌دهد. سپس، الگوریتم با استفاده از رابطه به‌روزرسانی $\theta_{k+1} = \theta_k - \alpha \nabla J(\theta_k)$، گامی به اندازه α در خلاف جهت گرادیان برمی‌دارد تا مقدار تابع هزینه کاهش یابد. پس از به‌روزرسانی پارامترها، مقدار خطای جدید (e_{rr}) به صورت نُرم اقلیدسیِ اختلاف مقادیر جدید و قدیم پارامترها محاسبه می‌شود.

این فرآیند تا زمانی تکرار می‌شود که شرط توقف ارضا گردد؛ یعنی یا مقدار خطای e_{rr} از حد ϵ تعیین شده کمتر شود (که نشان‌دهنده همگرایی الگوریتم است)، و یا تعداد تکرارها (k) به سقف مجاز $N_{\max}$ برسد تا از افتادن در حلقه‌های بی‌نهایت جلوگیری شود. در نهایت، بهترین مقادیر یافت شده برای پارامترها به عنوان خروجی بازگردانده می‌شوند.

الگوریتم ۱ الگوریتم گرادیان کاهشی

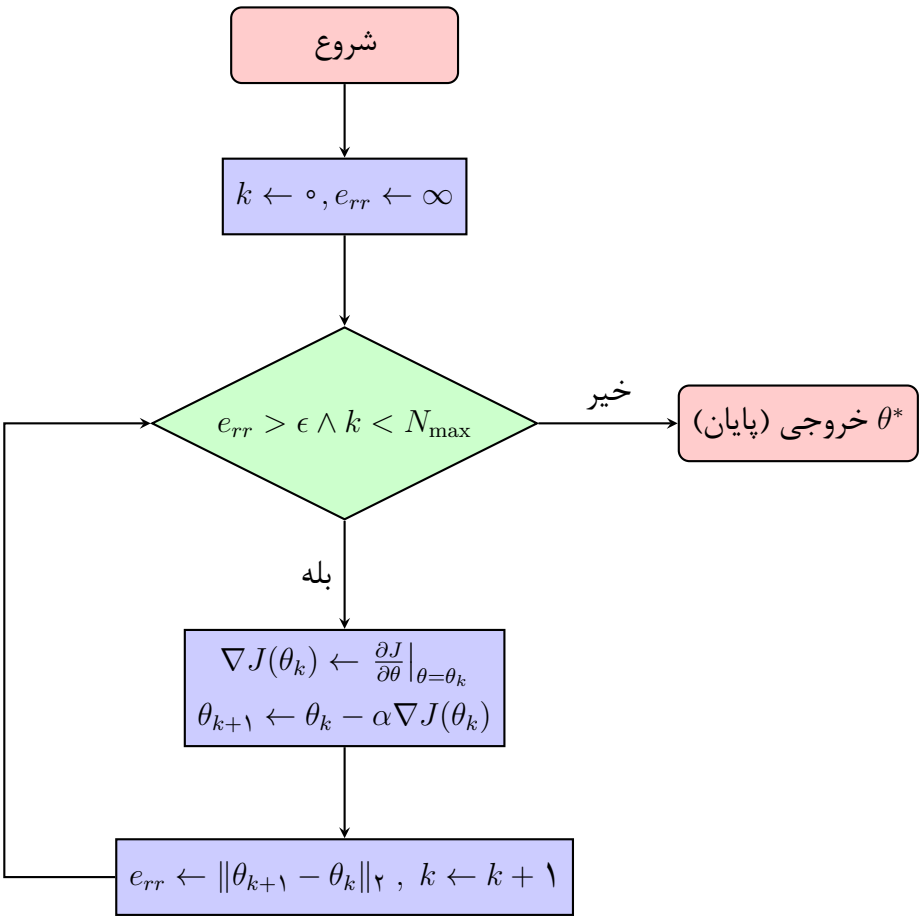
ورودی: پارامترهای اولیه θ_0، نرخ یادگیری α، حد تحمل ϵ، حداکثر تکرار $N_{\max}$
خروجی: پارامترهای بهینه‌شده θ^*

$e_{rr} \leftarrow \infty$
$k \leftarrow 0$
تا زمانی که $e_{rr} > \epsilon \wedge k < N_{\max}$ **انجام بده**
$\nabla J(\theta_k) \leftarrow \frac{\partial J}{\partial \theta}\Big|_{\theta=\theta_k}$
$\theta_{k+1} \leftarrow \theta_k - \alpha \nabla J(\theta_k)$
$e_{rr} \leftarrow ||\theta_{k+1} - \theta_k||_2$
$k \leftarrow k + 1$
پایان تا زمانی که
بازگردان θ_k

[10] Learning Rate

۲-۲-۲ روندنمای فرآیند

به منظور درک بهتر روند اجرای الگوریتم ۱، روندنما مربوطه در شکل ۲-۱ رسم شده است. این شکل به وضوح نشان می‌دهد که چگونه متغیرهای حالت و مقدار خطا در هر تکرار به‌روزرسانی می‌شوند تا زمانی که شرط توقف برآورده گردد.



شکل ۲-۱: فلوچارت فرآیند بهینه‌سازی و بررسی شرط توقف

فصل سوم

توابع زیان و ارزیابی مدل‌ها

۳-۱ مقدمه

در این فصل به بررسی انواع توابع زیان[۱]، روش‌های منظم‌سازی[۲] و ارزیابی مدل‌های یادگیری ماشین می‌پردازیم. انتخاب تابع زیان مناسب، تأثیر مستقیمی بر نحوه همگرایی الگوریتم و مقاومت آن در برابر نویز و داده‌های پرت[۳] دارد.

جدول ۳-۱ خلاصه‌ای از سه رابطه کلیدی را که در بخش‌های آتی این فصل مورد بحث و بررسی دقیق‌تر قرار می‌گیرند، نشان می‌دهد.

جدول ۳-۱: معرفی توابع زیان و روابط تحلیلی در ارزیابی مدل

نام رابطه	فرمول ریاضی
تابع هزینه رگرسیون ریج	$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$ (۳-۱)
تابع زیان هیوبر	$L_\delta(e) = \begin{cases} \frac{1}{2}e^2 & \text{اگر } \lvert e\rvert \le \delta \\ \delta\lvert e\rvert - \frac{1}{2}\delta^2 & \text{در غیر این صورت} \end{cases}$ (۳-۲)
معادله نرمال	$\theta = \left(X^TX + \lambda I\right)^{-1}X^Ty$ (۳-۳)

۳-۲ منظم‌سازی و رگرسیون ریج

هنگامی که مدل ما دچار بیش‌برازش[۴] می‌شود، یکی از راهکارهای مرسوم استفاده از تکنیک‌های منظم‌سازی است. همان‌طور که در رابطه (۳-۱) از جدول ۳-۱ مشاهده کردید، تابع هزینه رگرسیون ریج[۵] از دو بخش تشکیل شده است. بخش اول همان خطای میانگین مربعات است و بخش دوم یک عبارت جریمه[۶] بر مبنای نرم L_2 بردار وزن‌هاست.

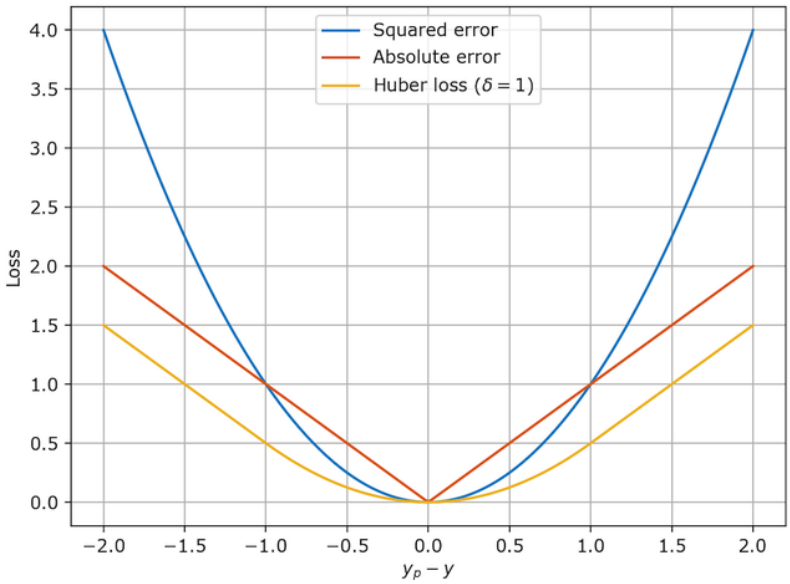
[1] Loss Functions
[2] Regularization
[3] Outliers
[4] Overfitting
[5] Ridge Regression
[6] Penalty Term

این جریمه که با ضریب λ کنترل می‌شود، از رشد بی‌رویه مقادیر پارامترهای θ_j جلوگیری می‌کند و در نتیجه پیچیدگی مدل را کاهش می‌دهد.

۳-۳ تابع زیان هیوبر

در بسیاری از مسائل رگرسیون، استفاده از خطای میانگین مربعات[۷] به عنوان تابع هزینه، مدل را به شدت نسبت به داده‌های پرت[۸] حساس می‌کند؛ زیرا خطاهای بزرگ به توان دو می‌رسند و جریمه سنگینی به مدل تحمیل می‌کنند.

برای مقابله با این مشکل، از تابع زیان هیوبر[۹] استفاده می‌شود.[۲] همان‌طور که در رابطه (۳-۲) از جدول ۳-۱ مشخص است، این تابع به صورت چندضابطه‌ای تعریف می‌شود. تابع هیوبر ترکیبی از بهترین ویژگی‌های خطای میانگین مربعات [۱۰] و خطای میانگین مطلق است. یک پارامتر تنظیم‌کننده به نام δ در این تابع وجود دارد که اگر اندازه خطا (یعنی e) کمتر یا مساوی δ باشد، تابع رفتار مربعی از خود نشان می‌دهد که باعث مشتق‌پذیری و همگرایی نرم‌تر در نزدیکی نقطه بهینه می‌شود. اما اگر خطا از δ بیشتر شود (که معمولاً نشان‌دهنده یک داده پرت است)، تابع رفتار خطی پیدا می‌کند و از رشد بی‌رویه گرادیان‌ها جلوگیری می‌نماید. همان‌طور که در شکل ۳-۱ مشاهده می‌شود، رفتار توابع زیان مختلف در برابر خطای پیش‌بینی کاملاً متفاوت است.



شکل ۳-۱: مقایسه رفتار توابع خطای مربعات ، خطای مطلق و زیان هیوبر ($\delta = ۱$).

[7] Mean Squared Error (MSE)
[8] Outliers
[9] Huber Loss
[10] Mean Absolute Error (MAE)

در این نمودار، محور افقی نشان‌دهنده میزان خطا (تفاضل مقدار پیش‌بینی‌شده و مقدار واقعی) و محور عمودی نشان‌دهنده مقدار زیان است. با بررسی نمودار می‌توان به نتایج زیر دست یافت:

- **خطای مربعات (منحنی آبی):** این تابع رشد سهموی دارد. برای خطاهای بزرگتر از ۱ یا کوچکتر از ۱−، مقدار زیان به شدت افزایش می‌یابد که نشان‌دهنده حساسیت بالای میانگین مربعات خطا به داده‌های پرت است.
- **خطای مطلق (منحنی قرمز):** این تابع به صورت خطی رشد می‌کند. شیب ثابت این تابع باعث می‌شود در برابر داده‌های پرت مقاوم‌تر باشد، اما مشتق‌ناپذیری آن در نقطه صفر می‌تواند در فرآیند بهینه‌سازی چالش‌برانگیز شود.
- **زیان هیوبر (منحنی زرد):** در بازه [۱−, ۱] رفتار منحنی مشابه تابع مربعات است که بهینه‌سازی را هموار می‌کند، اما در خطاهای بزرگتر، رفتار آن مشابه خطای مطلق می‌شود تا در برابر داده‌های پرت مقاومت کند.

۳-۴ معادله نرمال

علاوه بر روش‌های تکرارشونده مانند الگوریتم گرادیان کاهشی، برای یافتن پارامترهای بهینه (θ) در مسائل رگرسیون خطی می‌توان از یک راهکار تحلیلی و مستقیم به نام معادله نرمال[11] استفاده کرد.

با صفر قرار دادن مشتق تابع هزینه رگرسیون ریج نسبت به بردار θ، به رابطه (۳-۳) که در جدول ۳-۱ آورده شده است، دست می‌یابیم. در این رابطه X ماتریس ویژگی‌ها، y بردار مقادیر هدف و I ماتریس همانی است. اضافه شدن عبارت λI (مربوط به جریمه ریج) علاوه بر جلوگیری از بیش‌برازش، یک مزیت مهم دیگر نیز دارد: این عبارت تضمین می‌کند که ماتریس $(X^T X + \lambda I)$ همواره معکوس‌پذیر باشد، حتی اگر تعداد ویژگی‌ها از تعداد نمونه‌های آموزشی بیشتر باشد. با این حال، محاسبه معکوس ماتریس از نظر محاسباتی برای داده‌های بسیار بزرگ هزینه‌بر است و در چنین مواقعی استفاده از گرادیان کاهشی ترجیح داده می‌شود.

۳-۵ الگوریتم‌های بهینه‌سازی

در کنار انتخاب توابع زیان مناسب، نحوه کمینه‌سازی این توابع نیز از اهمیت بالایی برخوردار است. الگوریتم‌های بهینه‌سازی وظیفه دارند تا با به‌روزرسانی مکرر پارامترهای مدل، مقدار تابع زیان را به حداقل برسانند. انتخاب الگوریتم مناسب می‌تواند تأثیر چشمگیری بر سرعت همگرایی و کیفیت نهایی مدل داشته باشد.

[11] Normal Equation

در جدول ۳-۲ ، مقایسه‌ای بین چند الگوریتم پرکاربرد در یادگیری عمیق و ماشین ارائه شده است. همان‌طور که مشاهده می‌شود، الگوریتم‌های پیشرفته‌تری مانند آدام با وجود نیاز به حافظه کمی بیشتر، سرعت همگرایی بالاتری داشته و در برابر نویز داده‌ها مقاوم‌تر هستند.[۴]

جدول ۳-۲: مقایسه الگوریتم‌های بهینه‌سازی مختلف

نام الگوریتم	سرعت همگرایی	نیاز به حافظه	حساسیت به نویز
گرادیان کاهشی استاندارد	کند	کم	زیاد
گرادیان کاهشی تصادفی	متوسط	کم	بسیار زیاد
الگوریتم آدام	بسیار سریع	متوسط	کم

۳-۶ معیارهای ارزیابی مدل‌ها

پس از آموزش مدل با استفاده از توابع زیان و الگوریتم‌های بهینه‌سازی مناسب، گام بعدی ارزیابی عملکرد مدل بر روی داده‌های آزمون است. معیارهای ارزیابی به ما کمک می‌کنند تا میزان تعمیم‌پذیری مدل را سنجیده و در صورت نیاز، فرایند تنظیم ابرپارامترها را انجام دهیم. بسته به نوع مسئله (رگرسیون یا دسته‌بندی)، از معیارهای متفاوتی استفاده می‌شود.

۳-۶-۱ معیارهای ارزیابی در مسائل رگرسیون

در مسائل رگرسیون، هدف پیش‌بینی یک مقدار پیوسته است. بنابراین، معیارهای ارزیابی معمولاً بر اساس فاصله بین مقادیر واقعی و مقادیر پیش‌بینی‌شده بنا می‌شوند. دو معیار بسیار پرکاربرد در این زمینه عبارتند از:

میانگین مربعات خطا: این معیار میانگین مربعات اختلاف بین مقادیر واقعی و پیش‌بینی‌شده را محاسبه می‌کند. به دلیل توان دوم، این معیار خطاهای بزرگتر را با شدت بیشتری جریمه می‌کند.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{۳-۴}$$

میانگین قدر مطلق خطا: این معیار میانگین اندازه خطای مطلق را نشان می‌دهد و بر خلاف معیار قبلی، حساسیت کمتری نسبت به داده‌های پرت دارد.

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{۳-۵}$$

۳-۶-۲ معیارهای ارزیابی در مسائل دسته‌بندی

در مسائل دسته‌بندی، خروجی مدل یک کلاس گسسته یا احتمالی از تعلق به یک کلاس است. در اینجا معیارهای ارزیابی بر اساس تعداد پیش‌بینی‌های درست و نادرست تعریف می‌شوند.

دقت: ساده‌ترین معیار ارزیابی است که نسبت تعداد پیش‌بینی‌های صحیح به کل داده‌ها را نشان می‌دهد. اگرچه این معیار شهود خوبی ارائه می‌دهد، اما در صورت نامتوازن بودن کلاس‌ها معیار مناسبی نیست.

آنتروپی متقاطع: این معیار که اغلب به عنوان تابع زیان نیز استفاده می‌شود، برای ارزیابی خروجی‌های احتمالی مدل‌های دسته‌بندی (مانند رگرسیون لجستیک یا شبکه‌های عصبی) کاربرد فراوانی دارد. رابطه آن برای یک مسئله چندکلاسه به صورت زیر است:

$$\mathrm{CE} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{۳-۶}$$

که در آن C تعداد کلاس‌ها، y_i برچسب واقعی و $\hat{y}_i$ احتمال پیش‌بینی‌شده برای کلاس i است.

همانطور که در روابط (۳-۴) تا (۳-۶) مشاهده می‌شود، انتخاب معیار ارزیابی وابستگی مستقیمی به ماهیت داده‌ها و هدف نهایی مسئله دارد.

فصل چهارم

شبکه‌های عصبی عمیق

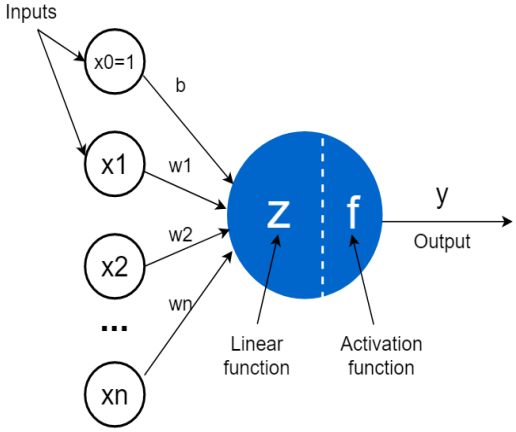
۴-۱ مقدمه

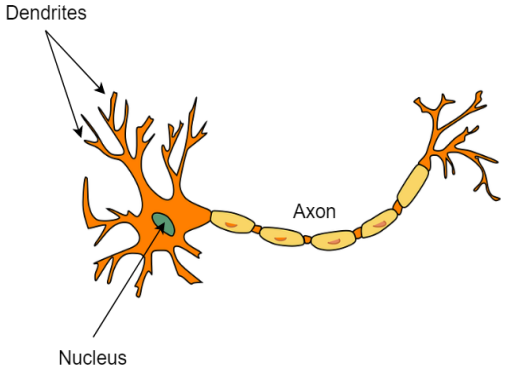
شبکه‌های عصبی مصنوعی[1] یکی از مهم‌ترین زیرشاخه‌های یادگیری ماشین هستند که ساختار و عملکرد آن‌ها از شبکه‌های عصبی بیولوژیکی مغز انسان الهام گرفته شده است[۱] این شبکه‌ها از تعداد زیادی واحدهای پردازشی ساده به نام یاخته عصبی[2] تشکیل شده‌اند که به صورت شبکه‌ای به هم متصل هستند. قدرت اصلی شبکه‌های عصبی مصنوعی در توانایی آن‌ها برای یادگیری الگوهای پیچیده و غیرخطی از درون داده‌ها نهفته است، که این ویژگی آن‌ها را برای حل مسائل پیچیده‌ای نظیر بینایی ماشین، پردازش زبان طبیعی و تشخیص گفتار به ابزاری بی‌بدیل تبدیل کرده است.

۴-۲ یاخته عصبی مصنوعی

واحد پایه‌ای و سازنده شبکه‌های عصبی مصنوعی، یاخته عصبی مصنوعی [3] نام دارد. برای درک بهتر نحوه عملکرد یاخته عصبی مصنوعی، می‌توان ساختار آن را با یک یاخته عصبی زیستی مقایسه کرد. همان‌طور که در شکل ۴-۱ مشاهده می‌شود، می‌توان اجزای یک یاخته عصبی طبیعی (شکل ۴-۱ (آ)) را با معادل‌های ریاضی و منطقی آن در یک یاخته عصبی مصنوعی (شکل ۴-۱ (ب)) تطبیق داد. در شبکه عصبی زیستی، دارینه‌ها[4] وظیفه دریافت سیگنال‌ها از یاخته‌های عصبی دیگر را بر عهده دارند که این بخش در یاخته عصبی مصنوعی معادل موج‌های ورودی است. هسته[5] یا بدنه سلول، پردازش موج‌ها را انجام می‌دهد که در مدل مصنوعی با تابع جمع‌کننده خطی و تابع فعال‌ساز جایگزین شده است. در نهایت، آسه[6] که سیگنال خروجی را به سایر سلول‌ها منتقل می‌کند، معادل خروجی نهایی یاخته عصبی مصنوعی است.

[1] Artificial Neural Networks (ANN)
[2] Neuron
[3] Perceptron
[4] Dendrites
[5] Nucleus
[6] Axon





زیرشکل (آ) ساختار یک یاخته عصبی بیولوژیکی شامل دندریت‌ها، هسته و آکسون

زیرشکل (ب) مدل ریاضی یک یاخته عصبی مصنوعی شامل ورودی‌ها، وزن‌ها، جمع‌کننده خطی و تابع فعال‌ساز

شکل ۴-۱: ساختار یاخته عصبی طبیعی در سیستم عصبی و مدل‌سازی یاخته عصبی مصنوعی

از نظر ریاضی، عملکرد یک یاخته عصبی مصنوعی را می‌توان با استفاده از رابطه (۴-۱) بیان کرد:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) = f(W^T X + b) \tag{۴-۱}$$

در این معادله:

- متغیر $X = [x_1, x_2, \ldots, x_n]^T$ نشان‌دهنده بردار ورودی‌ها است.
- متغیر $W = [w_1, w_2, \ldots, w_n]^T$ بردار وزن‌های متصل به هر ورودی را نشان می‌دهد که اهمیت هر موج ورودی را تعیین می‌کند.
- پارامتر b (سوگیری[7]) مقداری است که به مجموع خطی اضافه می‌شود تا انعطاف‌پذیری مدل را برای برازش بهتر داده‌ها افزایش دهد.
- تابع f همان تابع فعال‌ساز[8] است که ویژگی‌های غیرخطی را به خروجی مدل اضافه کرده و خروجی نهایی یعنی y را تولید می‌کند.

[7] Bias

[8] Activation Function

۴-۳ شبکه‌های عصبی چندلایه

با وجود اینکه یک یاخته عصبی مصنوعی به تنهایی می‌تواند مسائل ساده و خطی را حل کند، اما برای یادگیری الگوهای پیچیده در داده‌های واقعی، به شبکه‌ای از این یاخته‌های عصبی نیاز داریم. اتصال چندین یاخته عصبی به یکدیگر در لایه‌های مختلف، ساختاری را به نام شبکه‌های عصبی چندلایه[۹] شکل می‌دهد.

۴-۳-۱ ساختار لایه‌ها

در یک شبکه عصبی معیار، یاخته‌های عصبی در سه نوع لایه مختلف دسته‌بندی می‌شوند که هر کدام وظیفه خاصی بر عهده دارند:

- **لایه ورودی**[۱۰]: این لایه نقطه آغازین شبکه است و وظیفه دریافت داده‌های خام را بر عهده دارد. تعداد یاخته‌های عصبیِ این لایه دقیقاً برابر با تعداد ویژگی‌های[۱۱] داده‌های ورودی ما است و هیچ‌گونه پردازشی روی داده‌ها انجام نمی‌دهد.
- **لایه‌های پنهان**[۱۲]: این لایه‌ها بین ورودی و خروجی قرار دارند و قلب تپنده شبکه عصبی محسوب می‌شوند. وظیفه اصلی استخراج الگوها و ویژگی‌های پنهانِ داده‌ها بر عهده این بخش است. اگر یک شبکه عصبی بیش از یک لایه پنهان داشته باشد، به آن شبکه عصبی عمیق[۱۳] گفته می‌شود و فرآیند آموزش آن را یادگیری عمیق[۱۴] می‌نامند.
- **لایه خروجی**[۱۵]: نتایج نهاییِ پردازشِ شبکه، از این لایه به دست می‌آید. تعداد یاخته‌های عصبیِ لایه خروجی به نوع مسئله بستگی دارد؛ مثلاً در یک مسئله پیش‌بینی قیمت (رگرسیون[۱۶]) تنها یک یاخته عصبی دارد، اما در مسئله تشخیص اعداد ۰ تا ۹ (دسته‌بندی)، شامل ۱۰ یاخته عصبی خواهد بود.

۴-۳-۲ توابع فعال‌ساز رایج

همان‌طور که در رابطه (۴-۱) دیدیم، توابع فعال‌ساز برای ایجاد خاصیت غیرخطی در شبکه استفاده می‌شوند. بدون وجود این توابع، هرچند شبکه ما لایه‌های بسیار زیادی داشته باشد، باز هم خروجیِ

[9] Multilayer Perceptron (MLP)
[10] Input Layer
[11] Features
[12] Hidden Layers
[13] Deep Neural Network (DNN)
[14] Deep Learning
[15] Output Layer
[16] Regression

نهایی صرفاً یک ترکیب خطیِ ساده از ورودی‌ها خواهد بود و شبکه نمی‌تواند مسائل پیچیده دنیای واقعی را حل کند. در ادامه چهار مورد از پرکاربردترین توابع فعال‌ساز با بیانی ساده معرفی می‌شوند:

۱. تابع سیگموئید [۱۷]

این تابع، هر عدد ورودی را به یک مقدار خروجی در بازه (۰, ۱) فشرده می‌کند. به دلیل همین ویژگی، معمولاً از این تابع برای مسائلی استفاده می‌شود که نیاز به پیش‌بینی احتمال وقوع یک اتفاق داریم. رابطه ریاضی آن به شکل زیر است:

$$f(x) = \frac{۱}{۱ + e^{-x}} \tag{۴-۲}$$

۲. تابع تانژانت هذلولوی [۱۸]

عملکرد این تابع بسیار شبیه به تابع سیگموئید است، با این تفاوت که خروجی‌های آن در بازه (−۱, ۱) قرار می‌گیرند. این موضوع باعث می‌شود میانگین داده‌ها حول عدد ۰ متمرکز شود و معمولاً به یادگیری سریع‌تر شبکه کمک می‌کند:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{۴-۳}$$

۳. تابع واحد خطی یکسوساز [۱۹]

این تابع، امروزه محبوب‌ترین و پرکاربردترین تابع فعال‌ساز در شبکه‌های عمیق است. منطق این تابع بسیار ساده است: اگر ورودی مثبت باشد، خودِ آن عدد را برمی‌گرداند و اگر منفی باشد، خروجی را برابر صفر قرار می‌دهد. این سادگیِ ریاضی باعث افزایش چشمگیر سرعت محاسبات در رایانه [۲۰] می‌شود:

$$f(x) = \max(۰, x) \tag{۴-۴}$$

۴. تابع بیشینه هموار [۲۱]

این تابع منحصراً در لایه خروجی برای مسائل دسته‌بندی چندطبقه‌ای [۲۲] استفاده می‌شود. این تابع خروجیِ تمام یاخته‌های عصبی لایه آخر را به گونه‌ای تنظیم می‌کند که مجموع آن‌ها دقیقاً برابر با عدد ۱ شود. به این ترتیب، می‌توانیم خروجی هر یاخته عصبی را به عنوان درصد احتمال تعلق داده به آن

[17] Sigmoid
[18] Hyperbolic Tangent (Tanh)
[19] Rectified Linear Unit (ReLU)
[20] Computer
[21] Softmax
[22] Multi-class Classification

طبقه[۲۳] در نظر بگیریم:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{۵-۴}$$

که در رابطه بالا، متغیر K نشان‌دهنده تعداد کل طبقه‌ها است.

۴-۳-۳ توابع زیان

پس از آنکه ساختار شبکه عصبی و توابع فعال‌ساز مشخص شدند، نیازمند معیاری هستیم تا میزان خطای شبکه را در پیش‌بینی‌هایش ارزیابی کنیم. تابع زیان[۲۴] وظیفه محاسبه این خطا را بر عهده دارد.

همان‌طور که در فصل۳ به تفصیل مورد بررسی قرار گرفت و روابط ریاضی آن‌ها بیان شد، بسته به نوع مسئله از توابع متفاوتی استفاده می‌شود؛ در مسائل رگرسیون معمولاً از تابع خطای میانگین مربعات یعنی رابطه (۳-۴) و در مسائل دسته‌بندی از تابع زیان آنتروپی متقاطع یعنی رابطه (۳-۶) استفاده می‌شود. شبکه عصبی با استفاده از الگوریتم‌های بهینه‌سازی تلاش می‌کند تا مقدار این توابع را کمینه کرده و به بهترین پیش‌بینی دست یابد.

۴-۴ آموزش شبکه عصبی

اکنون که ساختار شبکه عصبی، توابع فعال‌ساز و معیار ارزیابی (تابع زیان) مشخص شدند، این پرسش مطرح می‌شود که شبکه چگونه مقادیر بهینه برای وزن‌ها (W) و سوگیری‌ها (b) را پیدا می‌کند تا خطای پیش‌بینی به حداقل برسد؟ فرآیند یادگیری در شبکه‌های عصبی عمیق طی یک چرخه تکرارشونده شامل دو مرحله اصلی انجام می‌شود: انتشار رو به جلو و پس‌انتشار خطا.

۴-۴-۱ انتشار رو به جلو و پس‌انتشار خطا

در مرحله **انتشار رو به جلو**[۲۵]، داده‌های ورودی وارد شبکه شده و لایه به لایه در شبکه جلو می‌روند تا در نهایت یک خروجی (پیش‌بینی) تولید شود. سپس با استفاده از تابع زیان، میزان خطای این پیش‌بینی نسبت به مقدار واقعی محاسبه می‌گردد.

پس از محاسبه خطا، مرحله **پس‌انتشار خطا**[۲۶] آغاز می‌شود. این الگوریتم که ستون فقرات یادگیری عمیق محسوب می‌شود[۷]، با استفاده از قاعده زنجیره‌ای در حسابان[۲۷]، مشتق (گرادیان) تابع زیان را

[23] Class
[24] Loss Function
[25] Forward Propagation
[26] Backpropagation
[27] Chain Rule

نسبت به تک‌تک وزن‌ها و سوگیری‌های موجود در شبکه از لایه آخر به سمت لایه اول محاسبه می‌کند. این گرادیان‌ها نشان می‌دهند که هر وزن چقدر در ایجاد خطای نهایی سهم داشته است و برای کاهش خطا باید در چه جهتی تغییر کند.

۴-۴-۲ الگوریتم‌های بهینه‌سازی

پس از محاسبه گرادیان‌ها در مرحله پس‌انتشار، شبکه برای به‌روزرسانی پارامترهای خود از الگوریتم‌های بهینه‌سازی[۲۸] استفاده می‌کند. پایه و اساس اکثر این الگوریتم‌ها، روش **گرادیان کاهشی**[۲۹] است. در این روش، پارامترهای شبکه در خلاف جهت شیب (گرادیان) تابع زیان حرکت می‌کنند تا به کمینه سراسری (کمترین میزان خطا) برسند.

قانون پایه به‌روزرسانی وزن‌ها در الگوریتم گرادیان کاهشی به صورت رابطه (۴-۶) تعریف می‌شود:

$$w_{new} = w_{old} - \alpha \frac{\partial L}{\partial w} \tag{۴-۶}$$

در این رابطه:

- متغیر w نشان‌دهنده وزنِ یاخته عصبی است.
- پارامتر L همان تابع زیان است.
- عبارت $\frac{\partial L}{\partial w}$ نشان‌دهنده مشتق جزئی (گرادیان) تابع زیان نسبت به وزن است.
- متغیر α (آلفا) **نرخ یادگیری**[۳۰] نام دارد. این پارامترِ بسیار مهم، اندازه گام‌های شبکه را در مسیر کاهش خطا تعیین می‌کند. اگر α خیلی بزرگ باشد، ممکن است الگوریتم از نقطه بهینه عبور کند و اگر خیلی کوچک باشد، فرآیند یادگیری بسیار کند خواهد شد.

امروزه در شبکه‌های عصبی عمیق از نسخه‌های پیشرفته‌تر و تطبیق‌پذیرترِ گرادیان کاهشی استفاده می‌شود. یکی از محبوب‌ترینِ این روش‌ها، بهینه‌ساز **آدام**[۳۱] است.[۴] الگوریتم آدام با ترکیب مزایای روش‌های مختلف، نرخ یادگیری را برای هر پارامتر به صورت جداگانه و پویا در طول زمان تنظیم می‌کند که این امر منجر به همگرایی سریع‌تر و دقیق‌تر شبکه در مسائل پیچیده می‌شود.

[28] Optimization Algorithms
[29] Gradient Descent
[30] Learning Rate
[31] Adam Optimizer

۴-۵ کاربردهای شبکه‌های عصبی عمیق

با افزایش توان پردازشی رایانه‌ها و دسترسی به حجم عظیم داده‌ها (کلان‌داده‌ها)، شبکه‌های عصبی عمیق توانسته‌اند در بسیاری از حوزه‌های علمی و صنعتی تحولات چشمگیری ایجاد کنند. انعطاف‌پذیری این شبکه‌ها در یادگیری الگوهای پیچیده غیرخطی، باعث شده است تا در طیف وسیعی از مسائل به کار گرفته شوند. در ادامه به برخی از مهم‌ترین کاربردهای این شبکه‌ها اشاره می‌شود:

- **امنیت شبکه و فضای سایبری**[۳۲]: یکی از کاربردهای بسیار مهم و رو به رشد یادگیری عمیق، تامین امنیت شبکه‌های کامپیوتری است. از این شبکه‌ها در سیستم‌های تشخیص نفوذ[۳۳] برای شناسایی رفتارهای غیرعادی و ترافیک مخرب استفاده می‌شود. شبکه‌های عصبی می‌توانند با تحلیل الگوهای ترافیک شبکه، حملاتی مانند منع سرویس توزیع‌شده[۳۴]، بدافزارها و فیشینگ را با دقت و سرعت بسیار بالایی نسبت به روش‌های سنتی شناسایی کنند.[۹]

- **بینایی ماشین و پردازش تصویر**[۳۵]: از تشخیص چهره در تلفن‌های هوشمند گرفته تا سیستم‌های بینایی در خودروهای خودران و همچنین تشخیص تومورها در تصاویر پزشکی (مانند تصویربرداری تشدید مغناطیسی[۳۶] و اشعه ایکس[۳۷])، همگی از دستاوردهای شبکه‌های عصبی عمیق (به‌ویژه شبکه‌های پیچشی) هستند.

- **پردازش زبان طبیعی**[۳۸]: درک و تولید زبان انسان توسط ماشین، ترجمه ماشینی (مانند مترجم گوگل)، دستیارهای صوتی هوشمند و چت‌بات‌های پیشرفته، همگی بر پایه معماری‌های عمیق شبکه‌های عصبی بنا شده‌اند.

[32] Cyber & Network Security
[33] Intrusion Detection Systems (IDS)
[34] Distributed Denial of Service (DDoS)
[35] Computer Vision
[36] MRI
[37] X-Ray
[38] Natural Language Processing (NLP)

فصل پنجم

جمع‌بندی و نتیجه‌گیری و پیشنهادات

در این فصل، پس از بررسی جامع مفاهیم پایه‌ای، مدل‌سازی‌های ریاضی، الگوریتم‌های بهینه‌سازی و ساختار شبکه‌های عصبی عمیق، به جمع‌بندی دستاوردها و نتایج حاصل از این مستند می‌پردازیم. همچنین با تکیه بر دانش نظری به‌دست‌آمده در فصول پیشین، پیشنهادهایی کاربردی و مسیرهایی نوین برای توسعه این پژوهش و انجام کارهای آینده ارائه خواهد شد.

۵-۱ جمع‌بندی و نتیجه‌گیری

هدف اصلی این مستند، ایجاد یک پیوند علمی و منطقی میان مفاهیم بنیادین ریاضیات، الگوریتم‌های محاسباتی و فناوری‌های نوین یادگیری ماشین بود. روند این پژوهش نشان داد که چگونه مسائلی که در نگاه اول صرفاً تئوری به نظر می‌رسند، می‌توانند پایه و اساس فناوری‌های پیچیده امروزی باشند. نتایج و دستاوردهای اصلی این پژوهش را می‌توان در محورهای زیر خلاصه نمود:

- **شناخت الگوهای پنهان در ساختارهای ریاضی:** در فصل نخست نشان داده شد که مسائلی نظیر تجزیه اعداد صحیح و توزیع اعداد اول که پایه و اساس سامانه‌های رمزنگاری و امنیت دیجیتال هستند، دارای رفتارهای هندسی معناداری (مانند مارپیچ اولام) می‌باشند. درک این الگوها پیش‌نیاز توسعه الگوریتم‌های هوشمند است.

- **اهمیت مدل‌سازی و بهینه‌سازی ریاضی:** در فصول دوم و سوم، چارچوب‌های تحلیلی و روش‌های بهینه‌سازی مورد ارزیابی قرار گرفتند. بررسی توابع زیان مختلف (مانند زیان هیوبر در برابر خطای میانگین مربعات) و الگوریتم‌هایی نظیر گرادیان کاهشی و آدام ثابت کرد که انتخاب درست معیار ارزیابی و روش بهینه‌سازی، نقشی حیاتی در مقاومت مدل در برابر داده‌های پرت و سرعت همگرایی آن دارد.

- **قدرت شبکه‌های عصبی در حل مسائل پیچیده:** در فصل چهارم با بررسی ساختار یاخته‌های عصبی مصنوعی و شبکه‌های چندلایه، نشان داده شد که این مدل‌ها به کمک توابع فعال‌ساز غیرخطی و الگوریتم پس‌انتشار خطا، توانایی فوق‌العاده‌ای در استخراج الگوهای پنهان داده‌ها دارند.

- **هم‌گرایی امنیت و هوش مصنوعی:** در نهایت، به عنوان یکی از مهم‌ترین نتایج، مشخص شد که معماری‌های عمیق می‌توانند به عنوان ابزاری قدرتمند برای تامین امنیت فضای سایبری به کار گرفته شوند. مدل‌های یادگیری عمیق با تحلیل ترافیک شبکه توانایی بالایی در شناسایی حملات پیچیده نظیر منع سرویس توزیع‌شده و تشخیص نفوذ دارند و می‌توانند جایگزین قدرتمندی برای روش‌های سنتیِ مبتنی بر امضا باشند.

به طور خلاصه، این مستند یک چارچوب نظری منسجم فراهم کرد که نشان می‌دهد گذار از امنیت سنتیِ مبتنی بر پیچیدگی‌های محاسباتی (مانند تجزیه اعداد) به سمت امنیت هوشمندِ مبتنی بر داده (شبکه‌های عصبی عمیق)، یک مسیر علمی ضروری و اجتناب‌ناپذیر است.

۵-۲ پیشنهادها برای کارهای آینده

با توجه به مباحث نظری مطرح شده و پتانسیل بالای شبکه‌های عصبی عمیق، موارد زیر به عنوان پیشنهادهایی برای توسعه و ادامه این پژوهش در کارهای آینده ارائه می‌گردد:

۱. **پیاده‌سازی عملی مدل‌های تشخیص نفوذ:** پیشنهاد می‌شود در پژوهش‌های آتی، مفاهیم تئوری این مستند به صورت عملی پیاده‌سازی شوند. به عنوان مثال، استفاده از یک مجموعه دادگان[۱] استاندارد ترافیک شبکه و آموزش یک شبکه عصبی عمیق برای تشخیص بلادرنگ[۲] حملات سایبری و مقایسه دقت[۳] آن با روش‌های کلاسیک می‌تواند بسیار ارزشمند باشد.

۲. **استفاده از یادگیری عمیق در تحلیل رمزنگاری[۴]:** با توجه به الگوهای بررسی شده در مارپیچ اولام، یکی از ایده‌های جذاب برای آینده، تغذیه این الگوهای بصری و هندسیِ اعداد اول به شبکه‌های عصبی (به‌ویژه شبکه‌های عصبی پیچشی[۵]) است تا مشخص شود آیا هوش مصنوعی قادر به کشف روابط جدیدی برای تسریع فرآیند تجزیه اعداد صحیح می‌باشد یا خیر.

۳. **بررسی معماری‌های پیشرفته‌تر شبکه‌های عصبی:** در این مستند تمرکز اصلی بر شبکه‌های عصبی چندلایه بود. پیشنهاد می‌شود برای داده‌های توالی‌دار مانند ترافیک شبکه که وابستگی زمانی دارند، از معماری‌های پیشرفته‌تری نظیر شبکه‌های عصبی بازگشتی[۶] یا حافظه طولانی کوتاه‌مدت[۷] استفاده شود.

۴. **ترکیب الگوریتم‌های فراابتکاری با یادگیری عمیق:** بررسی ترکیب الگوریتم‌های بهینه‌سازی تکاملی با روش‌هایی مانند آدام برای فرار از کمینه‌های محلی[۸] و بهبود فرآیند آموزش شبکه‌های عصبی، می‌تواند موضوع یک پژوهش مستقل و نوآورانه باشد.

[1] Dataset
[2] Real-time
[3] Accuracy
[4] Cryptanalysis
[5] CNN
[6] RNN
[7] LSTM
[8] Local Minima

منابع و مراجع

[1] Goodfellow, I, Bengio, Y, and Courville, A. Deep learning, mit press. available online. 2016.

[2] Hampel, Frank R. Introduction to huber (1964) robust estimation of a location parameter. In *Breakthroughs in Statistics: Methodology and Distribution*, pages 479–491. Springer, 1992.

[3] Hardy, Godfrey Harold and Wright, Edward Maitland. *An introduction to the theory of numbers*. Oxford university press, 1979.

[4] Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Rivest, Ronald L, Shamir, Adi, and Adleman, Leonard. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[6] Ruder, Sebastian. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[7] Rumelhart, David E, Hinton, Geoffrey E, and Williams, Ronald J. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[8] Stein, Myron L, Ulam, Stanislaw M, and Wells, Mark B. A visual display of some properties of the distribution of primes. *The American Mathematical Monthly*, 71(5):516–520, 1964.

[9] Xin, Yang, Kong, Lingshuang, Liu, Zhi, Chen, Yuling, Li, Yanmiao, Zhu, Hongliang, Gao, Mingcheng, Hou, Haixia, and Wang, Chunhua. Machine learning and deep learning methods for cybersecurity. *Ieee access*, 6:35365–35381, 2018.