

RAG Ingestion Verification Report

Source

Source ID: src_a681b098698a4d359705a5a881568f48

Source Type: SourceType.PDF

Source URI: C:\Users\Utkarsh Mishra\Downloads\Main Projects\RAG_QA\uploaded_files\test3.pdf

File Hash: 4d65784fdb345e0be14189df56dd01695b06324665456af0fe0d2f1036fac787

Ingested At: 2026-02-09 13:38:19.658246+00:00

Document

Document ID: doc_c3583aa90e4646119aca3c3f53c35998

Source ID: src_a681b098698a4d359705a5a881568f48

Document Hash: 222c20d56458308b5805f6a6a93fa49c7d109c723170a72e3bbc0ec94ea9c742

Version: 1

Created At: 2026-02-09 13:38:49.438015

Content Units

Unit 1 [image / vision]

Unit ID: unit_d5587b6be8b84617aa0c577a21a5f1a2

Page: 2

Content Hash: 42ab3629d7c8d82401d5b25525b4fd505998a9955c8d934dfe2d4a73799608a8

Image Path: storage\artifacts\images\src_a681b098698a4d359705a5a881568f48_p2_0.png

■ *Image could not be rendered*

Unit 2 [image / vision]

Unit ID: unit_33098ef8780942b8b5958cae5cccb08b

Page: 2

Content Hash: 4d4e4005c3d1b34afa87e3ae3e0cc12b57ffd3d9014bf639ce9e085422c17c1e

Image Path: storage\artifacts\images\src_a681b098698a4d359705a5a881568f48_p2_1.png

■ *Image could not be rendered*

Unit 3 [image / vision]

Unit ID: unit_f7d857ff106247a9a40d068d7d395bf6

Page: 2

Content Hash: 79fcfa5b35b55c109687dbd307136bcd7d0f43cc50baf268232051475bc967e

Image Path: storage\artifacts\images\src_a681b098698a4d359705a5a881568f48_p2_2.png

■ *Image could not be rendered*

Unit 4 [image / vision]

Unit ID: unit_df9d4d3bf5324d82a4ce40725676b922

Page: 3

Content Hash: 9f737dd86aaba9cd465523c70b0d3bacf2848e3c9bc813de140305b3a26fa5cc

Image Path: storage\artifacts\images\src_a681b098698a4d359705a5a881568f48_p3_0.png

■ *Image could not be rendered*

Unit 5 [text / ocr]

Unit ID: unit_824900ecf0f3497bb076bfd0f99ce3dd

Page: 1

Content Hash: 1430f8fbb5929ed562a10db343c27b319674f719a8ab1a14ac90195906139189

Page 1 of 22 1. INTRODUCTION TO CNN Convolutional Neural Networks (CNNs) are a special type of neural network designed to work well with image and spatial data (data arranged in a grid, like 2D images or even 1D time series). Instead of connecting every input pixel to every neuron (like a fully connected ANN), CNNs use small filters (kernels) that scan across the image to detect patterns. This approach allows the network to automatically learn useful visual features: o Early layers > detect edges and simple shapes. o Middle layers > detect textures or parts of objects. o Deeper layers > detect whole objects or complex patterns. CNNs are parameter-efficient compared

to ANN: the same small filter is reused across the whole image (called weight sharing). This makes them faster to train and better at understanding images than a standard fully connected network.

1.1 Summary CNNs are neural networks specialized for grid-like data (like images) that learn visual features automatically and are far more efficient than fully connected ANNs. 1.2 Problem with ANN on images An image of size $64 \times 64 \times 3$ (RGB) has 12,288 pixels. A single fully connected layer with just 100 neurons would need $12,288 \times 100 = 1.2\text{M}$ weights > huge, slow, prone to overfitting. CNN fixes this by using small filters that are reused across the image instead of learning a separate weight per pixel. 1.3 Why CNN (Convolutional Neural Networks) Uses small filters/kernels (e.g., 3×3 , 5×5) that slide over the image. The same small set of weights is reused across the entire image (weight sharing). Learns local spatial patterns first (edges > shapes > objects). Far fewer parameters, trains faster, generalizes better.

Unit 6 [table | ocr]

Unit ID: unit_cfb2115ad70c407586c38d7b6f3e7ccc

Page: 1

Content Hash: b9fb7d29549c34701f7bf0e5f9c26fa0dfb97b8191800610e16f69093c0a7f89

s) that scan across the image to detect patterns. This approach allows the network to automatically identify objects in the image based on learned features. The process starts by scanning the image pixel by pixel, extracting local features at each pixel, and then comparing them against a database of known patterns. If a match is found, the system identifies the object; if not, it continues to scan the rest of the image. This iterative process ensures that all objects in the image are detected and identified.

Unit 7 [text | ocr]

Unit ID: unit_440d96bf18a949478b9cb6f67a4459fd

Page: 2

Content Hash: 31e51e5148cd43aef8ee8f567d7ffe8d9b3c6299432d2387182ac221e03456a4

Page 2 of 22 2. HOW CNN WORKS — VISUAL INTUITION Think of an image as a grid of numbers (pixel values). A filter/kernel is a small grid of weights (e.g., 3x3) that slides (convolves) across the image: Input Image fe.g., 6x6 Ge & ae te HOS Or we ®orRPNH® © PN HF @ @ He Ho OH fai camel} mah a Goma} mak opm} 3x3 Filter e = The filter slides over the image, multiplying and summing values > creates a feature map. e One filter may detect horizontal edges, another vertical edges, etc. e Stacking many filters helps the network learn different features automatically. 2.1 CNN Feature Learning Hierarchy e _ Layer 1: Learns edges (horizontal, vertical, diagonal). e Layer 2: Learns textures, corners. e —_ Layer 3+: Learns shapes and objects (faces, wheels, etc.). 2.2 One-liner for interviews: “CNNs use small filters that slide over an image to detect patterns. Early layers find simple edges; deeper layers combine them into complex shapes and objects — all with far fewer parameters than a fully connected ANN.” 2.3 ANN v CNN ANN CNN Fully connected — huge number of weights Convolution filters + very few weights Ignores spatial info Exploits local spatial structure Easily overfits on images Generalizes well 2.4 One-liner for Interviews “ANNs treat every pixel independently and explode in parameter count, while CNNs share small filters across the image, preserving spatial structure and using far fewer weights.”

Unit 8 [table | ocr]

Unit ID: unit_691fc2806658497eb93547a6db21820f

Page: 2

Content Hash: 6e9acf7bba0deada5e4797a71a22e024664043e1b7bf28112eda4c26cdaf529

Unit 9 [text | ocr]

Unit ID: unit_9a89e7a122bf420cb77425f1f9e183fa

Page: 3

Content Hash: e78ede2c18f51a944d29eb7454145487e36a0d96a148c07f0b1535baafbc78f39

Page 7 of 22 Step 2 — Move 1 step right Next 3x3 patch: [[2, 3, 0], [1, 2, 3], [2, 1, 0]] Multiply with filter F: $(2^*1) + (3^*0) + (0^*-1) + (1^*1) + (2^*0) + (3^*-1) + (2^*1) + (1^*0) + (0^*-1) = (2+0+0) + (1+0-3) + (2+0+0) = 2$ So, Output [0,1] = 2. You repeat this sliding process until the filter has scanned the whole 5x5 image. Since we used: e = Input = 5x5 e = Filter = 3x3 e §=6©Stride=1 e = Padding = Valid (no padding) «© The Output feature map size = 3x3. @ What do negative / zero / positive numbers in the feature map mean? When a filter slides over the image: e Each output number is the dot product (multiplication + sum) between the filter and the small patch of the image. e It's basically a score for "how well this patch matches the pattern" the filter has learned. Positive (large)|/Patch matches the filter's pattern strongly. Patch doesn't match the pattern much. Patch matches the opposite of the filter's pattern. ® Example: e If the filter learned to detect a vertical edge (bright on left, dark on right): o Positive > strong vertical edge found. o Zero > no vertical edge. o Negative > opposite edge (dark left, bright right). ® Putting it together e You choose how many filters and their size (e.g., 32 filters of size 3x3). e The network learns what each filter detects during training (no need to hand-design).

Unit 10 [table | ocr]

Unit ID: unit_03ada75ad72f4d2a921dcf3937b33f5

Page: 3

Content Hash: 0d07b2e1d80d6d74360f15bbbe7492759fec2ee2b05c4805b59ad35e44f6de1c

Multiply	with	filter	F:	(2^*1)	+	(3^*0)	+	(O^*-1)	+	(1^*1)	+
----------	------	--------	----	----------	---	----------	---	-----------	---	----------	---

age:	e	Each	output	number	is	the	dot	product	(multiplication	+	sum)	between
tical	edge	found.	o	Zero	>	no	vertical	edge.	o	Negative	>	opposite
e	network	learns	what	each	filter	detects	during	training	(no	need	to	hand-design).