

## 6. Testing and Validation Artefacts

Comprehensive testing and validation are essential for demonstrating reliability, safety, and performance in a regulated medical AI system. All tests use only public or fully anonymised datasets (no real patient data), aligning with UK GDPR, MHRA GMLP, and NHS digital standards. Artefacts are placed in `/tests/` and `/docs/validation/` for assessor review, with commit history showing iterative improvements.

### Key Components

- **Testing Framework:** pytest with coverage (pytest-cov) – unit tests for individual modules (anonymisation, ViT inference, LLM drafting), integration tests for pipeline end-to-end.
- **Datasets Used:**
  - Chest X-rays: MIMIC-CXR-JPG (PhysioNet), PadChest, NIH ChestX-ray14.
  - General/Other: TCIA collections (e.g., LIDC-IDRI for lung nodules).
  - Reports: IU X-ray or MIMIC-CXR paired reports for ground truth.
- **Metrics:**
  - ViT Analysis: Accuracy, Sensitivity, Specificity, AUC-ROC (per-class and macro-averaged).
  - Report Drafting: BLEU-4, ROUGE-L, METEOR (compared to reference reports).
- **Pilot Simulations:** Timed runs on sample cases (manual transcription vs AI-assisted editing) using public images.
- **Error Handling:** Structured logging (Python logging module) capturing invalid inputs, model failures, timeouts.

### Industry-Level Example Diagrams and Visuals

**Figure 43: Sample pytest unit and integration test results with coverage report** (Standard output showing passed/failed tests.)

```
~% 1 ~/unit_testing/project
~/unit_testing/project
unit_testing > pytest
=====
test session starts =
platform darwin -- Python 3.11.2, pytest-7.3.1, pluggy-1.0.0
rootdir: /Users/avel.docquin/unit_testing/project
collected 2 items

tests/test_math_utils.py F.
[100%]

=====
FAILURES =====
test_divide_positive_numbers

def test_divide_positive_numbers() -> None:
    """Test that divide returns the correct result when given two integers."""
>     assert divide(1, 2) == 0.6
E     assert 0.5 == 0.6
E       +  where 0.5 = divide(1, 2)

tests/test_math_utils.py:7: AssertionError
=====
short test summary info
FAILED tests/test_math_utils.py::test_divide_positive_numbers - assert 0.5 == 0.6
===== 1 failed, 1 passed in 0.02s =====

~/unit_testing/project
unit_testing > []
```

Python Unit Tests with pytest Guide | Avel Docquin | Medium

Suggested filename: fig43-pytest-results-example.pdf

Pytest vs. Unittest: Which Is Better? | The PyCharm Blog

Suggested filename: fig44-pytest-verbose-output.pdf

```

~/PycharmProjects/Personal/pytest-code-coverage-example ➜ master
$ coverage report -m
Name          Stmts  Miss  Cover  Missing
-----  

bank_app/__init__.py      0     0  100%  

bank_app/core.py        41     0  100%  

tests/__init__.py       0     0  100%  

tests/unit/__init__.py   0     0  100%  

tests/unit/conftest.py  32     0  100%  

tests/unit/test_bank_app.py 30     0  100%  

-----  

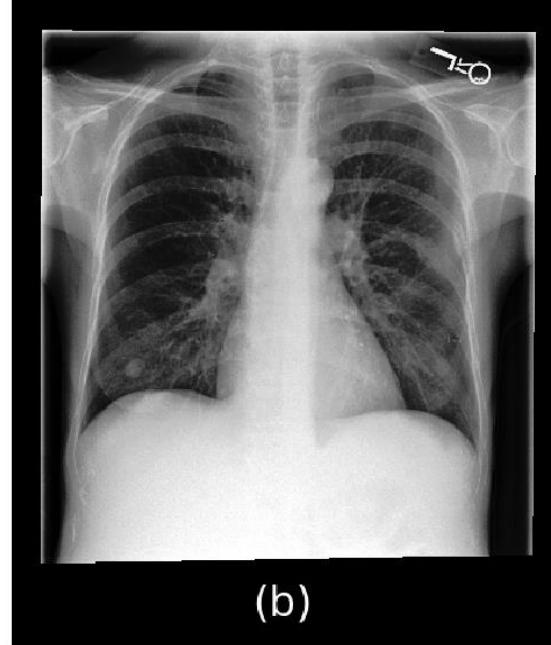
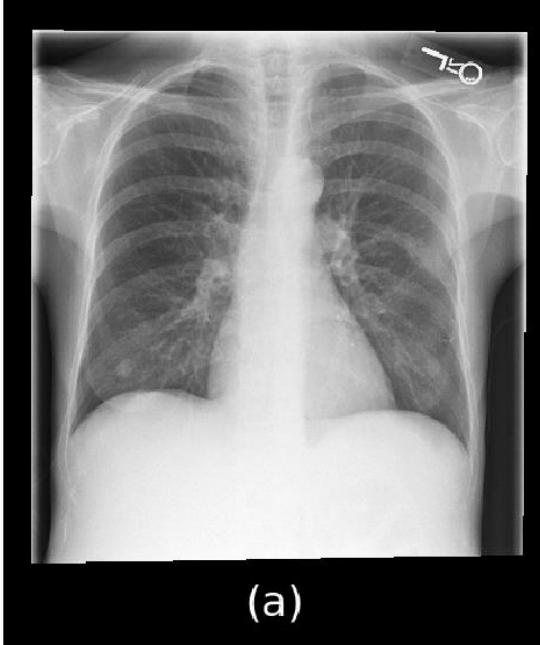
TOTAL            103     0  100%

```

How To Generate Beautiful & Comprehensive Pytest Code Coverage ...

Suggested filename: fig45-pytest-coverage-report.pdf

**Figure 46: Sample chest X-ray images from MIMIC-CXR dataset used for validation (Public anonymised examples for testing.)**



MIMIC-CXR-JPG, a large publicly available database of labeled ...

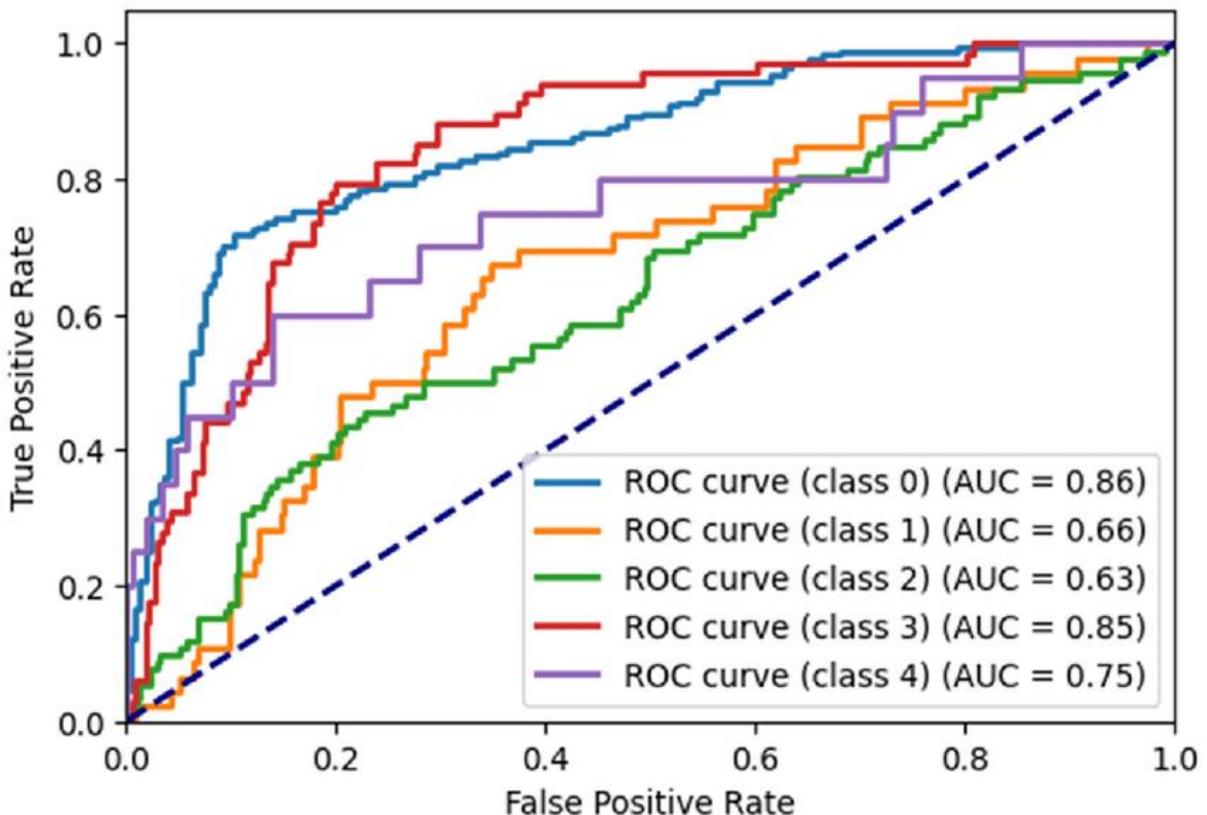
Suggested filename: fig46-mimic-cxr-samples.pdf

	Chest X-rays											
DenseNet-121	LO	0.816	FE	0.856	PE	0.849	NF	0.743	NF	0.612	FE	0.705
	Pn	0.784	C	0.565	A	0.582	Pn	0.094	LO	0.119	SD	0.346
	NF	0.743	A	0.550	SD	0.570	LO	0.082	Pn	0.115	C	0.344
	PE	0.336	E	0.446	C	0.480	A	0.063	A	0.079	A	0.342
	PI	0.260	SD	0.421	C	0.499	C	0.049	C	0.027	LO	0.237
	C	0.235	LO	0.343	LO	0.316	E	0.027	PE	0.044	PI	0.193
	A	0.226	PI	0.217	Pn	0.167	F	0.023	F	0.041	F	0.182
	SD	0.146	EC	0.112	EC	0.099	PE	0.015	LL	0.030	EC	0.090
	E	0.090	Ce	0.110	Co	0.092	SD	0.014	Pt	0.022	Co	0.083
	EC	0.055	F	0.075	Pt	0.065	LL	0.014	E	0.021	Pt	0.080
	Co	0.055	LL	0.022	NF	0.011	Co	0.013	EC	0.000	NF	0.075
	LL	0.051	PO	0.016	LL	0.021	FC	0.009	Co	0.020	LL	0.035
	F	0.029	F	0.012	F	0.017	Pt	0.008	SD	0.014	F	0.026
	PO	0.019	NF	0.010	PO	0.014	PO	0.006	PO	0.011	PO	0.020
DenseNet-KG	PI	0.857	FE	0.860	LO	0.808	NF	0.909	NF	0.834	PF	0.716
	LO	0.294	A	0.455	E	0.779	C	0.136	LO	0.281	A	0.450
	PE	0.247	C	0.381	PE	0.694	LO	0.132	Pn	0.173	SD	0.427
	Pn	0.241	SD	0.355	A	0.453	Pn	0.131	C	0.113	C	0.341
	C	0.277	F	0.310	SD	0.420	A	0.166	A	0.106	LO	0.312
	A	0.223	LO	0.393	C	0.528	E	0.062	SD	0.066	E	0.243
	SD	0.143	PI	0.145	Co	0.298	SD	0.036	PE	0.055	Pn	0.166
	NF	0.136	Co	0.070	Pn	0.164	F	0.028	F	0.051	Co	0.105
	E	0.099	EC	0.065	EC	0.087	PE	0.027	LL	0.039	Pt	0.096
	Co	0.072	Pt	0.051	Pt	0.048	LL	0.030	PI	0.036	EC	0.091
	EC	0.063	NF	0.010	LL	0.024	Co	0.017	Co	0.034	LL	0.036
	LL	0.045	LL	0.016	NF	0.019	EC	0.014	Pt	0.031	F	0.018
	F	0.041	F	0.011	F	0.017	Pt	0.010	Pt	0.029	NF	0.018
	PO	0.024	PO	0.011	PO	0.017	PO	0.008	PO	0.016	PO	0.017
VSE-GCN(22)	PI	0.923	FE	0.959	A	0.878	NF	0.969	F	0.829	PE	0.884
	A	0.367	A	0.925	E	0.704	Pn	0.163	NF	0.887	C	0.872
	LO	0.246	PI	0.989	LO	0.699	LO	0.102	LO	0.165	A	0.414
	F	0.196	C	0.484	PE	0.536	A	0.067	Pn	0.143	SD	0.413
	SD	0.187	F	0.355	Cu	0.470	C	0.056	C	0.141	E	0.345
	Pn	0.174	LO	0.313	C	0.434	F	0.030	A	0.136	LO	0.311
	NF	0.135	EC	0.103	SD	0.333	Pn	0.030	PE	0.107	Pn	0.184
	EC	0.055	Co	0.092	EC	0.076	LL	0.023	PE	0.090	EC	0.099
	Co	0.051	Pt	0.089	Pt	0.066	Co	0.016	SD	0.055	Co	0.090
	PE	0.046	NF	0.050	NF	0.056	SD	0.015	EC	0.027	NF	0.057
	LL	0.029	LL	0.032	LL	0.026	EC	0.014	LL	0.026	PI	0.032
	F	0.028	F	0.028	F	0.023	Pt	0.013	PI	0.024	F	0.029
	PO	0.016	PO	0.022	PO	0.016	PO	0.008	PO	0.011	PO	0.022

Diagnostic results of six samples from MIMIC-CXR test set. Three ...

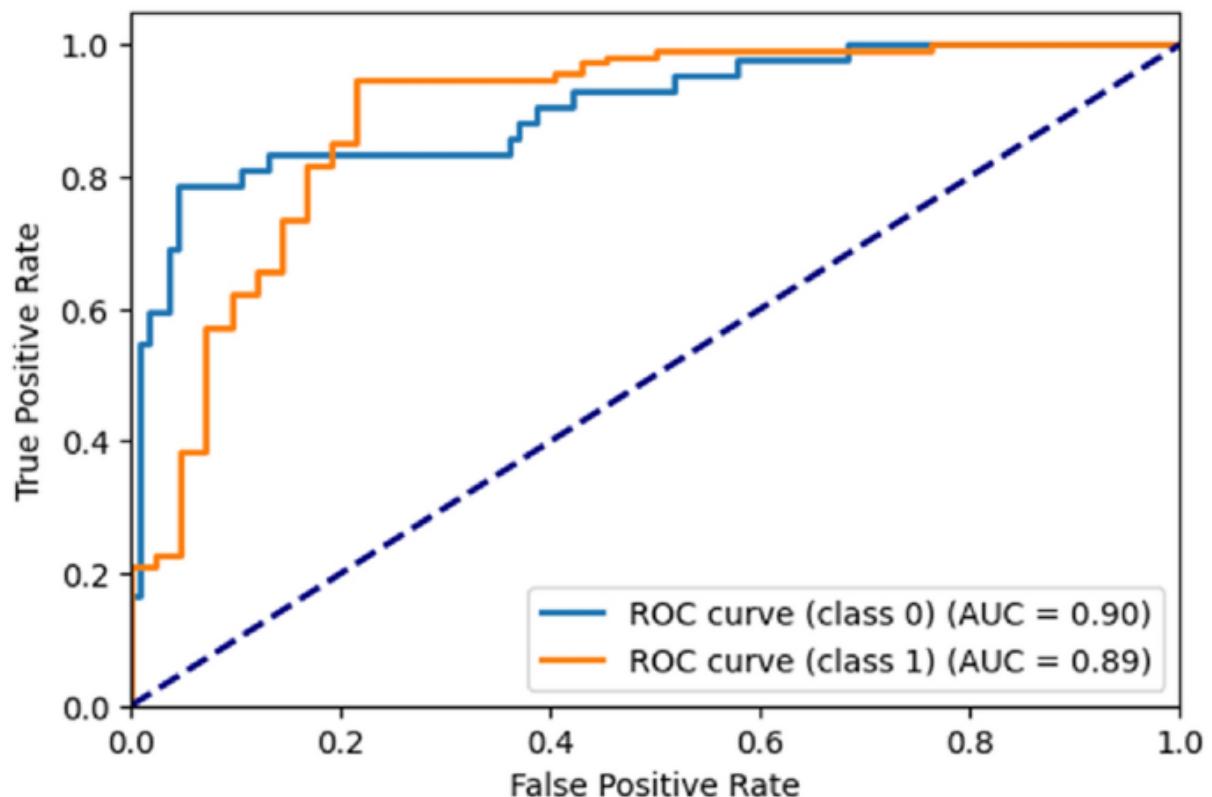
Suggested filename: fig47-mimic-cxr-diagnostic-samples.pdf

**Figure 48: ROC curves with AUC values for Vision Transformer models in medical imaging (Performance metrics for classification tasks.)**



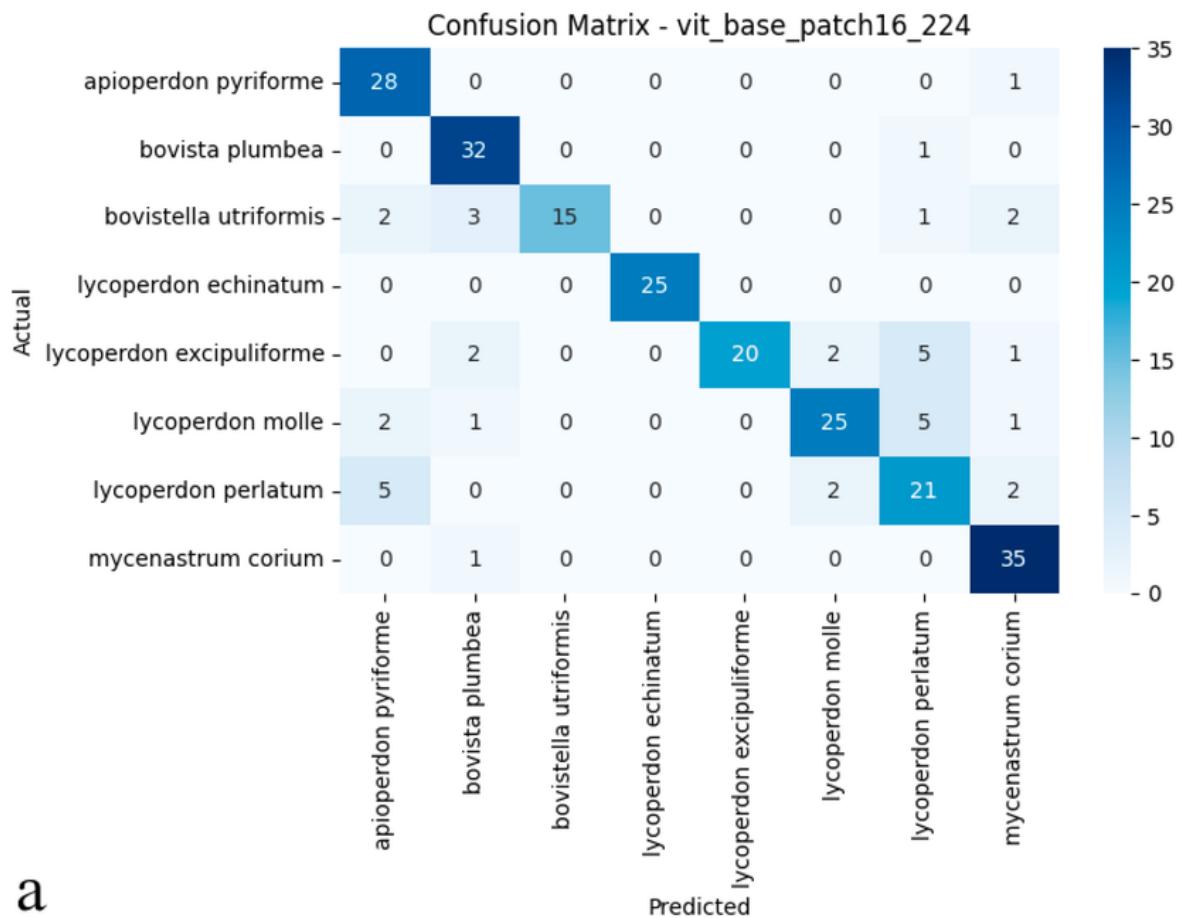
ROC curve generated by the proposed ViT model for RetinaMNIST ...

Suggested filename: fig48-vit-roc-curve.pdf



Implementing vision transformer for classifying 2D biomedical ...

Suggested filename: fig49-vit-biomedical-roc.pdf



a) Confusion matrix and (b) ROC curves with AUC values of the ...

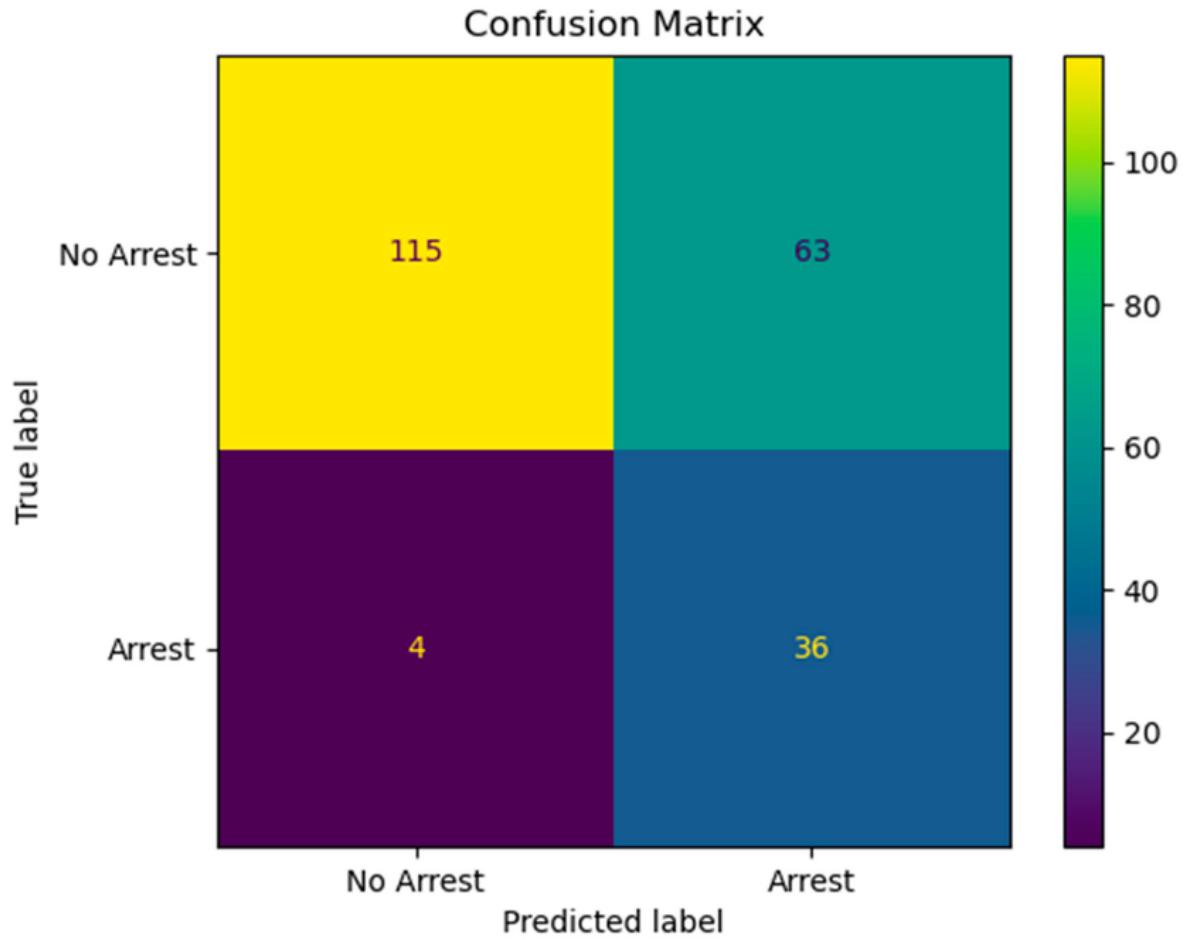
Suggested filename: fig50-vit-confusion-roc.pdf

**Figure 51: Confusion matrix showing sensitivity and specificity in medical AI evaluation (Diagnostic performance breakdown.)**

		Predicted Class		Sensitivity $\frac{TP}{(TP + FN)}$	Specificity $\frac{TN}{(TN + FP)}$	Accuracy $\frac{TP + TN}{(TP + TN + FP + FN)}$			
		Positive							
Actual Class	Positive	True Positive (TP) Type II Error							
	Negative	False Positive (FP) Type I Error		True Negative (TN)					
	Precision $\frac{TP}{(TP + FP)}$	Negative Predictive Value $\frac{TN}{(TN + FN)}$		Accuracy $\frac{TP + TN}{(TP + TN + FP + FN)}$					

terminology - What is the best way to remember the difference ...

Suggested filename: fig51-confusion-matrix-sens-spec.pdf



Confusion matrix: the model had a sensitivity of 90% and a ...

Suggested filename: fig52-medical-confusion-matrix.pdf

**Figure 53: BLEU and ROUGE score tables for LLM medical report generation evaluation (NLG metrics for draft quality.)**

Metric	Value Mean (std)	Interpretation
BLEU Score	0.82 (0.19)	High n-gram overlap with the reference text, indicating strong <b>word- and phrase-level</b> similarity.
ROUGE-1 F-measure	0.94 (0.08)	Excellent unigram recall, showing that most individual words match the reference text ( <b>word-level</b> ).
ROUGE-2 F-measure	0.87 (0.17)	Strong bigram overlap, reflecting the model's ability to capture <b>phrase-level</b> coherence.
ROUGE-L F-measure	0.92 (0.11)	High similarity in the longest common subsequence, suggesting well-preserved <b>sentence-level</b> structure.
METEOR	0.92 (0.11)	Incorporates <b>synonymy and word order alignment</b> , indicating semantically accurate and fluent descriptions.
BERTScore_F1	0.99 (0.02)	Extremely high <b>semantic similarity</b> based on contextual embeddings, showing alignment in meaning beyond surface-level text.

## Summary of text description evaluation metrics (BLEU, ROUGE ...)

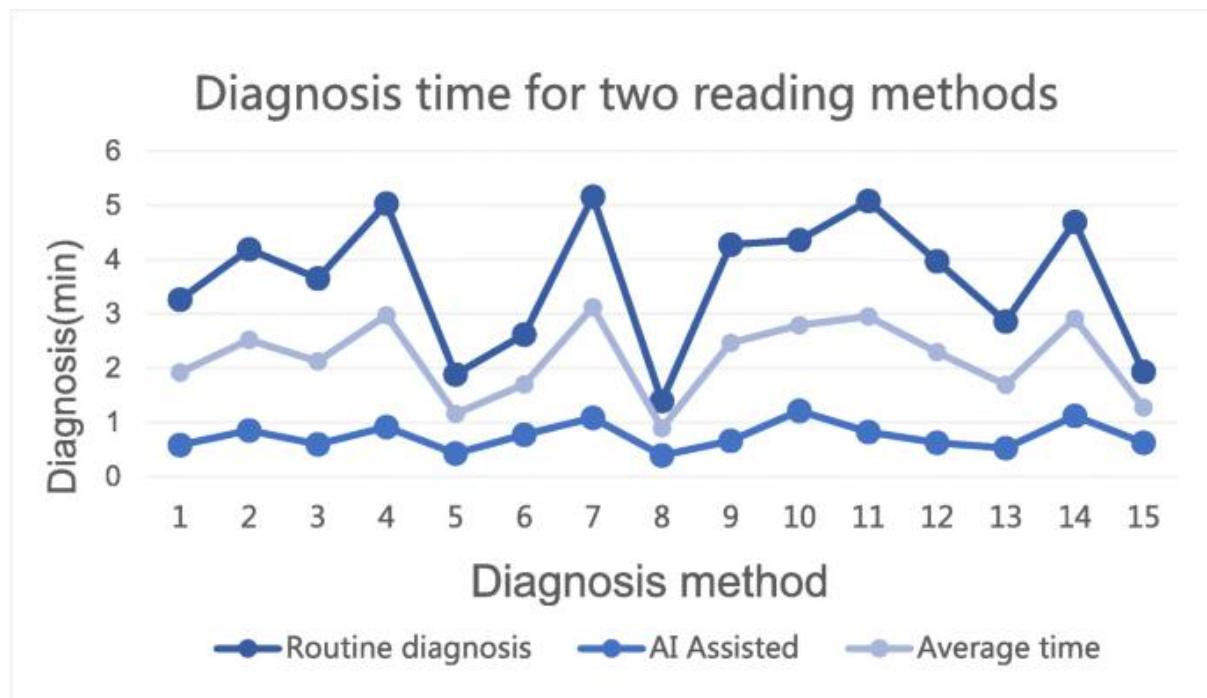
Suggested filename: fig53-bleu-rouge-table.pdf

Healthcare Categories & Subcategories	Dataset	Evaluation Metric	Model					
			Large		Medium		Small	
			GPT-4o	Gemini 1.5 Pro	Llama-3-70B-Instruct	GPT-4o-mini	Qwen-2.5-7B-Instruct	Phi-3.5-mini-instruct
<b>Clinical Decision Support</b>								
Supporting Diagnostic Decisions	MedCalc-Bench *	Exact Match	0.188	0.124	0.112	0.156	0.091	0.01
	CLEAR-AD	Exact Match	0.635	0.558	0.865	0.673	0.856	0.087
Planning Treatments	MT-Samples	BertScore-F1	0.722	0.702	0.717	0.709	0.708	0.725
Predicting Patient Risks and Outcomes	Medec	MedecFlagAcc	0.536	0.526	0.534	0.531	0.496	0.508
	EHRShot *	Exact Match	0.467	0.559	0.704	0.902	0.806	0.231
	MedQA *	Exact Match	0.91	0.84	0.848	0.832	0.734	0.679
Providing Clinical Knowledge Support	Medbullets	Exact Match	0.701	0.477	0.617	0.571	0.406	0.192
	MedAlign *	BertScore-F1	0.78	0.772	0.781	0.779	0.738	0.708
	ADHD-Behavior	Exact Match	0.837	0.761	0.641	0.806	-	-
	ADHD-MedEffects	Exact Match	0.945	0.731	0.925	0.756	-	-
<b>Clinical Note Generation</b>								
Documenting Patient Visits	DischargeMe *	BertScore-F1	0.754	0.762	0.751	0.758	0.751	0.755
	ACI-Bench	BertScore-F1	0.833	0.828	0.827	0.828	0.818	0.806
Recording Procedures	MT-Samples Procedures	BertScore-F1	0.713	0.706	0.705	0.706	0.704	0.723
Documenting Diagnostic Reports	MIMIC-RSS *	BertScore-F1	0.77	0.826	0.775	0.787	0.815	0.81
Documenting Care Plans	NoteExtract	BertScore-F1	0.769	0.772	0.782	0.769	0.743	0.761
<b>Patient Communication and Education</b>								
Providing Patient Education Resources	MedicationQA	BertScore-F1	0.756	0.714	0.747	0.755	0.735	0.725
Delivering Personalized Care Instructions	PatientInstruct	BertScore-F1	0.751	0.744	0.758	0.754	0.751	0.755
Patient Provider Messaging	MedDialog *	BertScore-F1	0.755	0.748	0.76	0.754	0.744	0.668
Enhancing Understanding in Health Communication	MedConInfo *	Exact Match	0.772	0.74	0.789	0.738	-	-
Facilitating Patient Engagement and Support	MEDIOA	BertScore-F1	0.774	0.75	0.769	0.775	0.778	0.777
Routine Health Monitoring	MentalHealth	BertScore-F1	0.821	0.797	0.802	0.819	0.787	0.39
<b>Medical Research Assistance</b>								
Conducting Literature Research	PubMedQA *	Exact Match	0.69	0.084	0.786	0.654	0.55	0.476
Analyzing Clinical Research Data	EHR-SQL	EHRSQLReAns	0.11	0.21	0.07	0.13	0	0.05
Recording Research Processes	BMT-Status	Exact Match	0.741	0.782	0.836	0.836	-	-
Ensuring Clinical Research Quality	RaceBias	Exact Match	0.874	0.82	0.599	0.76	0.587	0.144
Managing Research Enrollment	N2C2 CT	Exact Match	0.841	0.318	0.438	0.764	0.461	0.372
<b>Administration and Workflow</b>								
Scheduling Resources and Staff	HospiceReferral *	Exact Match	0.656	0.745	0.768	0.708	-	-
Overseeing Financial Activities	MIMIC-IV Billing Code *	Micro-F1	0.346	0.186	0.197	0.263	0.028	0.091
Organizing Workflow Processes	ClinicReferral	Exact Match	0.896	0.859	0.783	0.899	-	-
Care Coordination and Planning	CDI-QA *	Exact Match	0.589	0.466	0.563	0.579	-	-
	ENT-Referral *	Exact Match	0.611	0.576	0.363	0.594	-	-

## Holistic Evaluation of Large Language Models for Medical ...

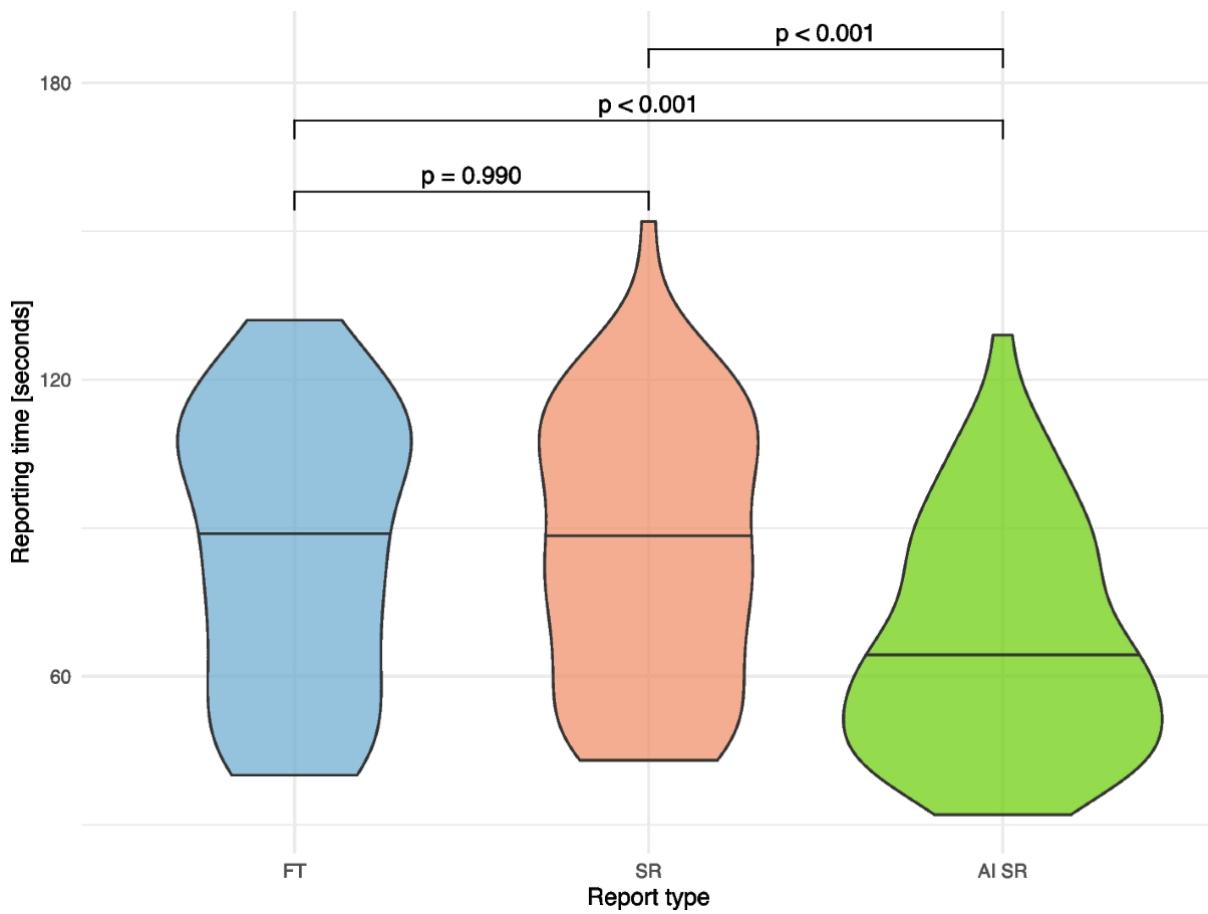
Suggested filename: fig54-llm-medical-metrics.pdf

**Figure 55: Time comparison charts for manual vs AI-assisted radiology reporting**  
(Simulated workflow efficiency gains.)

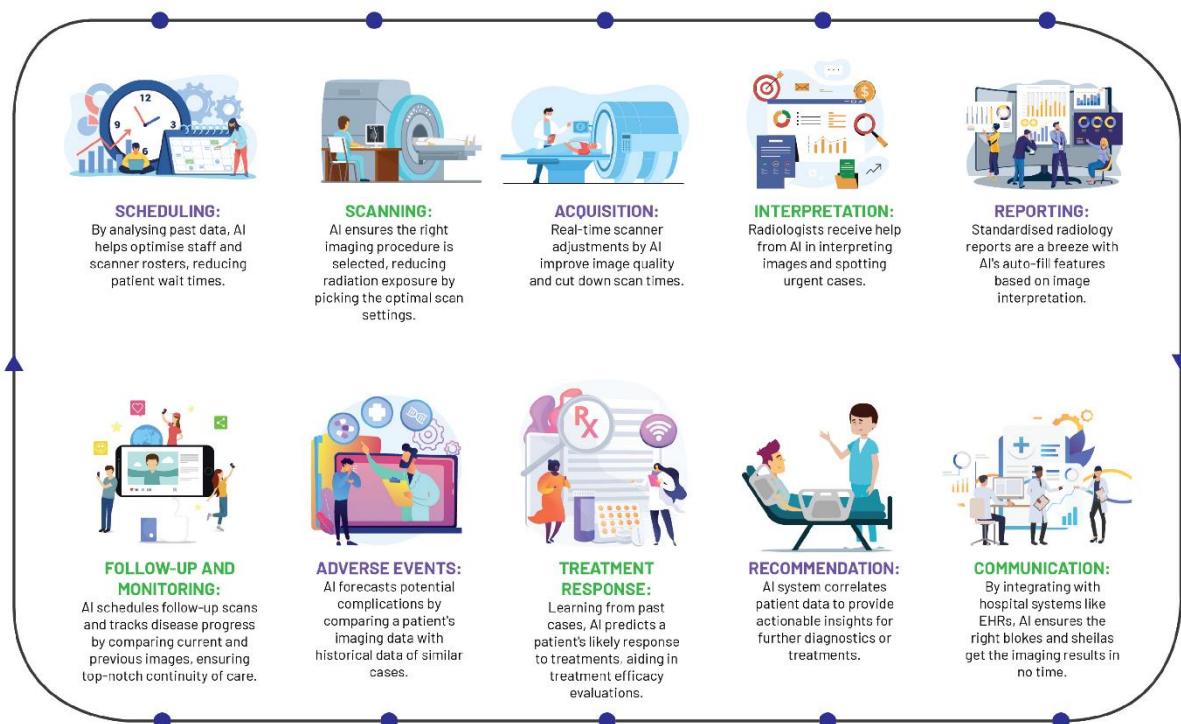


## RETRACTED ARTICLE: The value of artificial intelligence and ...

Suggested filename: fig55-ai-reporting-time-comparison.pdf



A novel reporting workflow for automated integration of artificial ...



## Redefining Radiology: A Review of Artificial Intelligence ...

Suggested filename: fig57-ai-efficiency-bar-chart.pdf

**Figure 58: Python logging and error handling examples** (Structured logs for failures.)

The screenshot shows the Spyder Python 3.7 IDE interface. In the top-left pane, there is an 'Editor' window titled 'handler.py' containing Python code for setting up a logger and writing log messages to a file. In the bottom-right pane, there is a 'debug.log - Notepad' window displaying the contents of the log file.

```
# -*- coding: utf-8 -*-
"""
Created on Tue Nov 27 09:38:09 2018
@author: IEUser
"""

import logging
logger = logging.getLogger(' Example _Log')
logger.setLevel(logging.DEBUG)
fh = logging.FileHandler('debug.log')
fh.setLevel(logging.DEBUG)
formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
fh.setFormatter(formatter)
logger.addHandler(fh)
logger.addHandler(fh)

logger.debug ('Debug Message')
logger.info ('Info Message')
logger.warning('Warning')
logger.error('Error Occured')
logger.critical('Critical Error')

```

debug.log - Notepad

```
2018-11-29 00:46:25,256 - Example_Log - DEBUG - Debug Message
2018-11-29 00:46:25,256 - Example_Log - INFO - Info Message
2018-11-29 00:46:25,256 - Example_Log - WARNING - Warning
2018-11-29 00:46:25,256 - Example_Log - ERROR - Error Occured
2018-11-29 00:46:25,256 - Example_Log - CRITICAL - Critical Error
```

## The Python logging module: How logging to file works - IONOS

Suggested filename: fig58-python-error-log.pdf

### Evidence Artefacts for Repository

- /tests/: test\_anonymizer.py, test\_vit\_inference.py, test\_report\_generator.py, test\_pipeline\_integration.py.
- Results: test\_report.html (pytest-html output), coverage reports.
- Validation docs: Tables of metrics (CSV/Markdown), timed simulation logs.
- Logs: error\_log\_sample.txt (anonymised entries).

These artefacts provide verifiable evidence of robust development, performance, and error resilience, supporting claims of workflow improvement in pilot simulations.