Task 2 & Task 3 Report

Task 2 — Embedding Model Design

For Task 2, the goal was to design a compact CNN that maps multi-coil complex MRI images (stacked real/imag channels from Task 1) into a fixed-size embedding vector suitable for contrastive learning.

Model Architecture (Layer by Layer):

1. Stem: LazyConv2d (out=32, stride=2, kernel=3) → GroupNorm → GELU
2. Stage 1: Conv2d(32→64, stride=2) → GroupNorm → GELU
3. Stage 2: Conv2d(64→128, stride=2) → GroupNorm → GELU
4. Stage 3: Conv2d(128→256, stride=2) → GroupNorm → GELU
5. Head: GlobalAveragePooling → Linear(256→256) → optional MLP head → L2 Normalization

We used GroupNorm instead of BatchNorm for robustness with small batch sizes, and GELU activations for smooth non-linearities. LazyConv2d was chosen in the stem to adapt automatically to any coil count. The design is deliberately simple and production-friendly.

Why This Architecture?

While alternatives such as ResNet-18 or ConvNeXt could have been used, the compact CNN offers a balance of simplicity, clarity, and sufficient performance for synthetic phantom data. It achieved perfect Recall@1 in validation. References consulted include ResNet, GroupNorm, GELU, and Triplet Loss literature (He et al. 2016, Wu & He 2018, Hendrycks & Gimpel 2016, Schroff et al. 2015).

Embedding Space

The embedding vectors are in R^256 and L2-normalized to lie on the unit hypersphere. This ensures cosine similarity and Euclidean distance are equivalent. Triplet loss enforces small distances for positive pairs and large distances for negatives. Empirically, d_pos ~0.1–0.2 and d_neg ~1.1–1.25, with Recall@1 consistently 100%.

Task 3 — Training with Contrastive Learning

The model was trained using the RandomPhantomTripletDataset and Triplet Loss.

Steps taken:

1. Created training dataset with two transform lists:
2.
    1. List A: 4× acceleration, 8% center fraction + Normalize + ToCompatibleTensor + Augmentation
    2. List B: 8× acceleration, 4% center fraction + Normalize + ToCompatibleTensor + Augmentation
3. Validation dataset with same transforms but without augmentation, with offset=len(train_ds) and deterministic=True.

4. Training loop: forward pass through model, compute Triplet Margin Loss (Euclidean), optimizer step with AdamW.
5. Validation loop: compute val loss, d_pos, d_neg, viol%, and Recall@1 each epoch.
6. Saved curves (PNG + CSV) and best checkpoint.

Experimental Results

Initial experiments showed fast convergence and near-zero validation loss early on. To make the experiment more reliable, we scaled dataset size (train=4000, val=1000), increased image size to 192, batch size to 32, and margin to 0.3. This led to stronger separation: d_pos ≈ 0.09–0.14, d_neg ≈ 1.1–1.25, viol% < 0.5%, and Recall@1 = 100%.

Compliance with Instructions

All required criteria were satisfied:

1. Two transform lists with specified undersampling fractions.
2. Validation dataset uses same transforms but no augmentation, offset, and deterministic mode.
3. Triplets used as required; embeddings normalized and compared with Euclidean distance.
4. Training and validation loops implemented with loss + metrics per epoch.
5. Curves and checkpoints saved for analysis.