Semantic Furniture Feature Extraction

by

decorAID

Semantic Furniture Feature Extraction

Find furnitures to realise your dream home

Click on furnitures in your image to view similar furnitures.

FortyTwo

Valencia 3 Seater...

FortyTwo

Cerny 4-Seater...

Breakdown of problem

How to search for similar furnitures?

↓

Clustering & feature extraction

Data collection

Categories

Product Page

Iteration 1: issues

Can use a better model

Fully Connected layers
with Relu activation function

256   128

ResNet
(freezed)

Better Grouping
- Sub-category is
  not a good
  measure of
  similarity

ResNet
(freezed)

256   128

Fully Connected layers
with Relu activation function

L1 Distance   Sigmoid

Binary Cross Entropy Loss

L1 distance + sigmoid
- Not differentiable
- Positive and unbounded
- Doesn't make sense to feed
  into Sigmoid

Binary Cross Entropy Loss
- Classification loss function
- Punished high and low L1
  distances, but that's not what
  is important

Iteration 2

Fully Connected layers

Use Blip as a base model

BLIP (Frozen)

BLIP (Frozen)

BLIP (Frozen)

Triplet Loss

Use triplet loss
- Relative distances matter more

Use manual grouping
- Ground truth
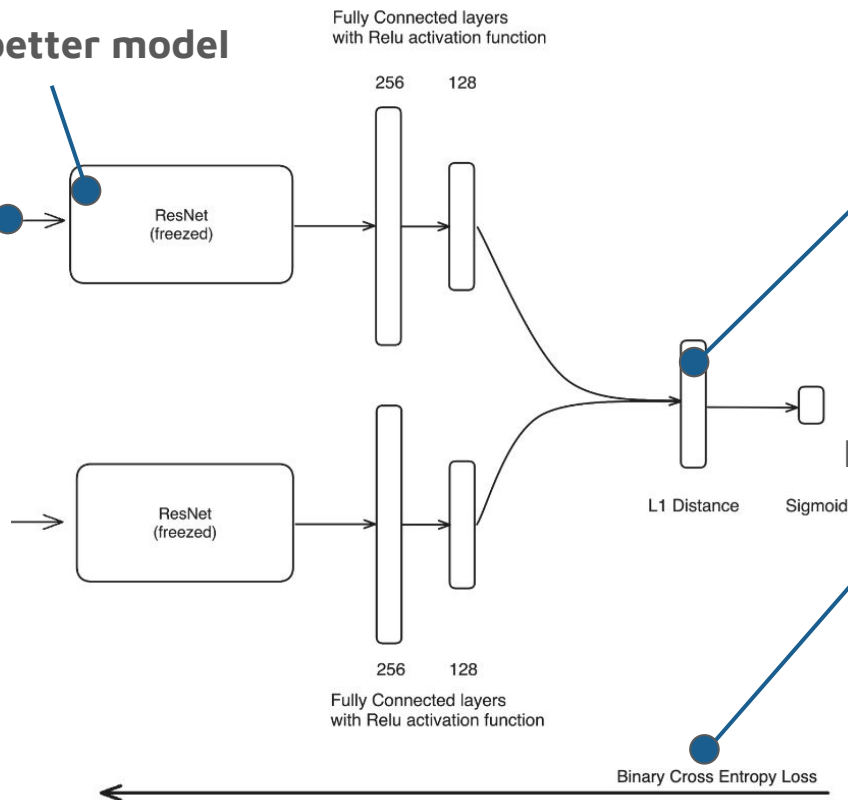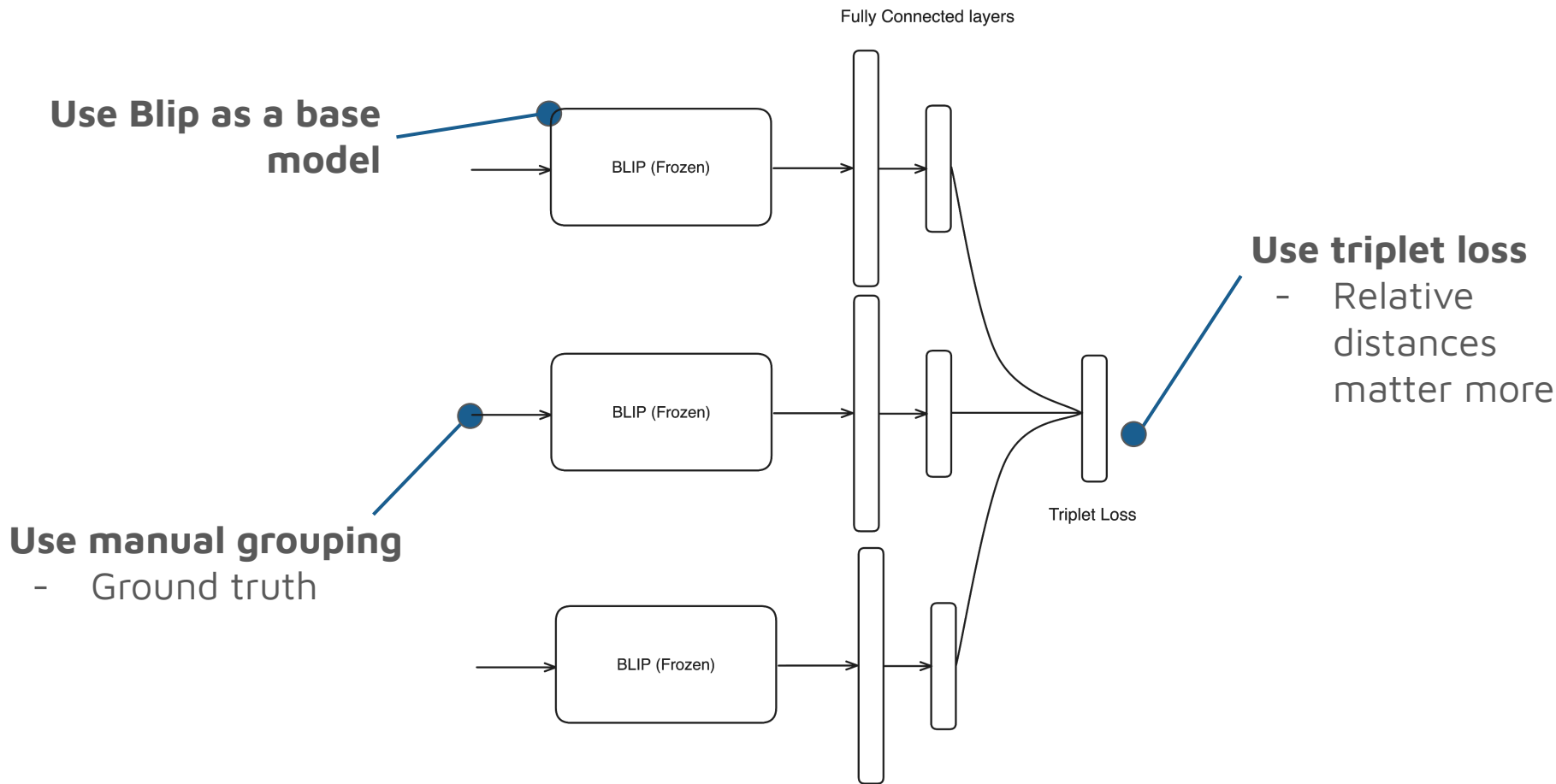
Manual grouping of clusters

- Group based on Form, Pattern and material

- Consistent interpretation of considerations

- Forms a solid ground truth

Splitting of dataset
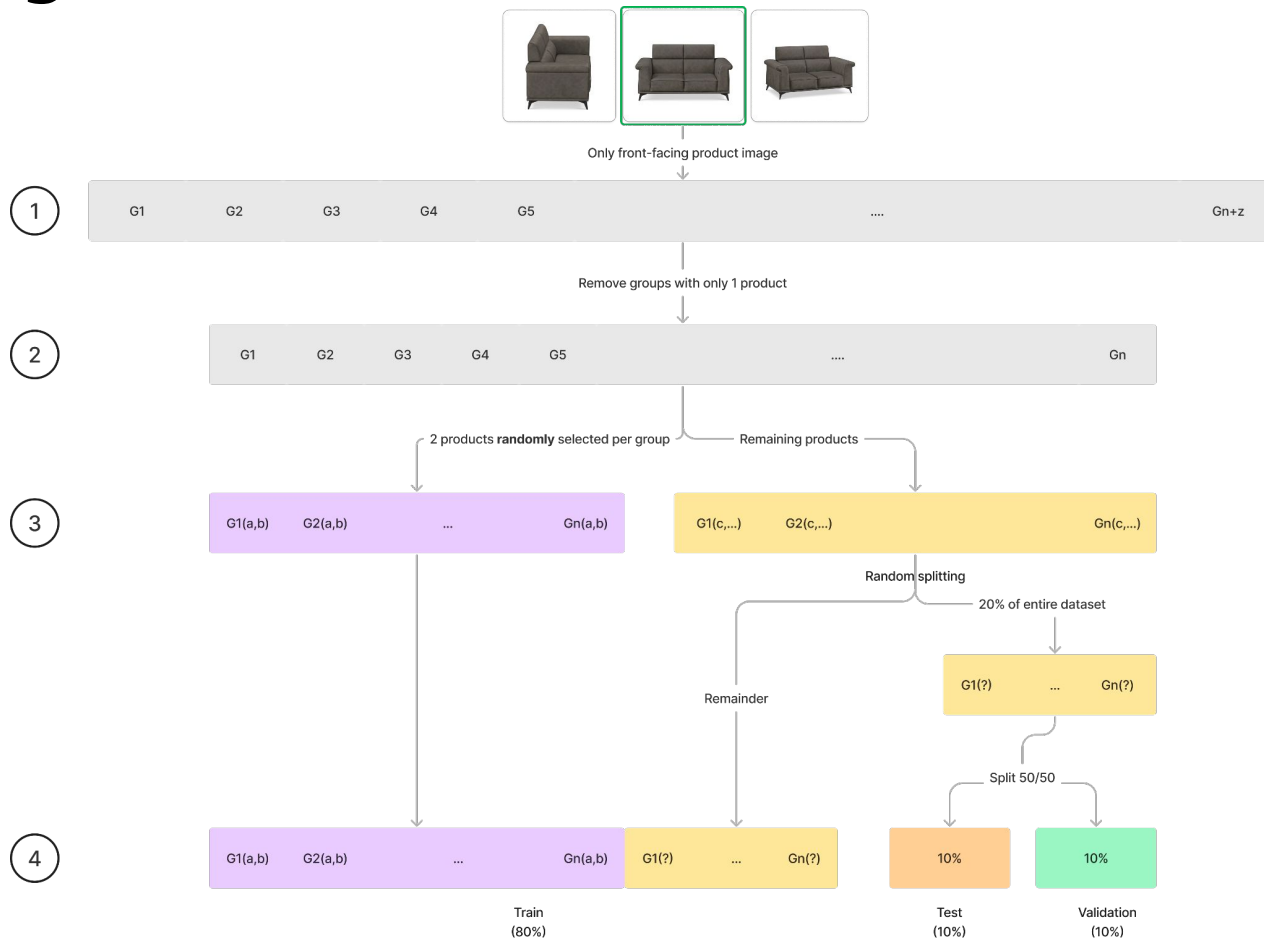
Generation of triplets
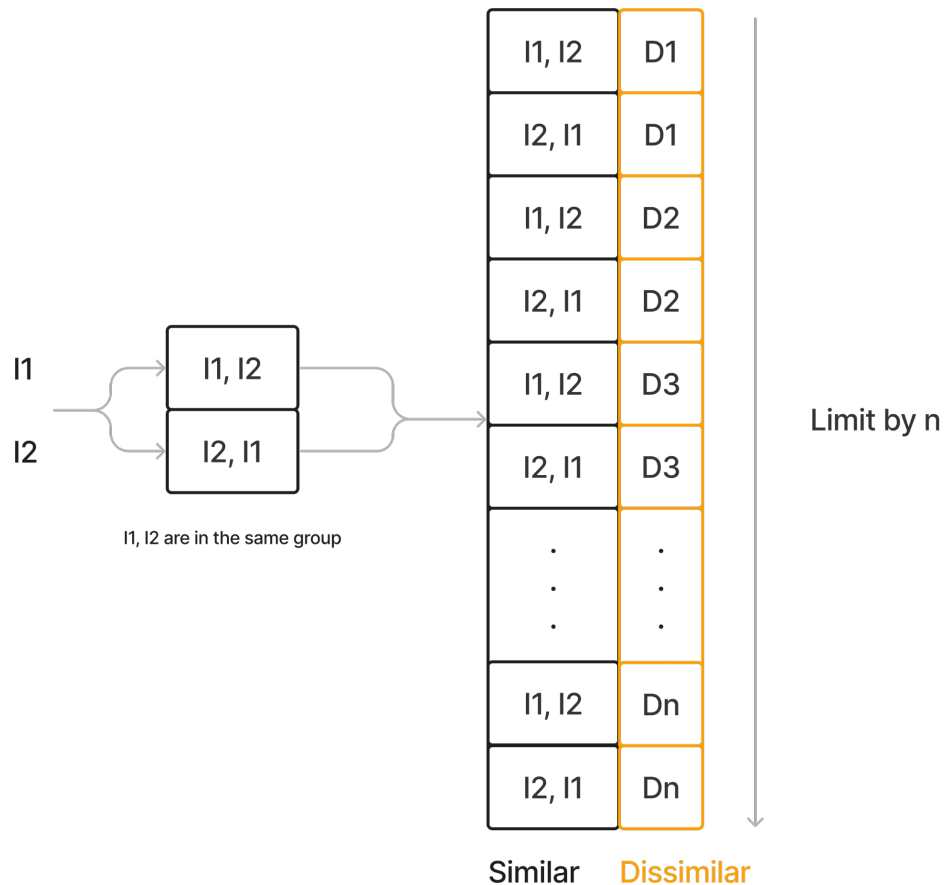
Evaluation Intuition

- Given an image, we want to measure if it is clustered among other products of the same group

- We know the trained model has some baseline understanding of the training data and has clustered them

- We can use the training data to form cluster centroids

- Given an evaluation image, we can then predict a product's group by its nearest cluster centroid

Getting centroids

Centroid

Furniture image from training dataset
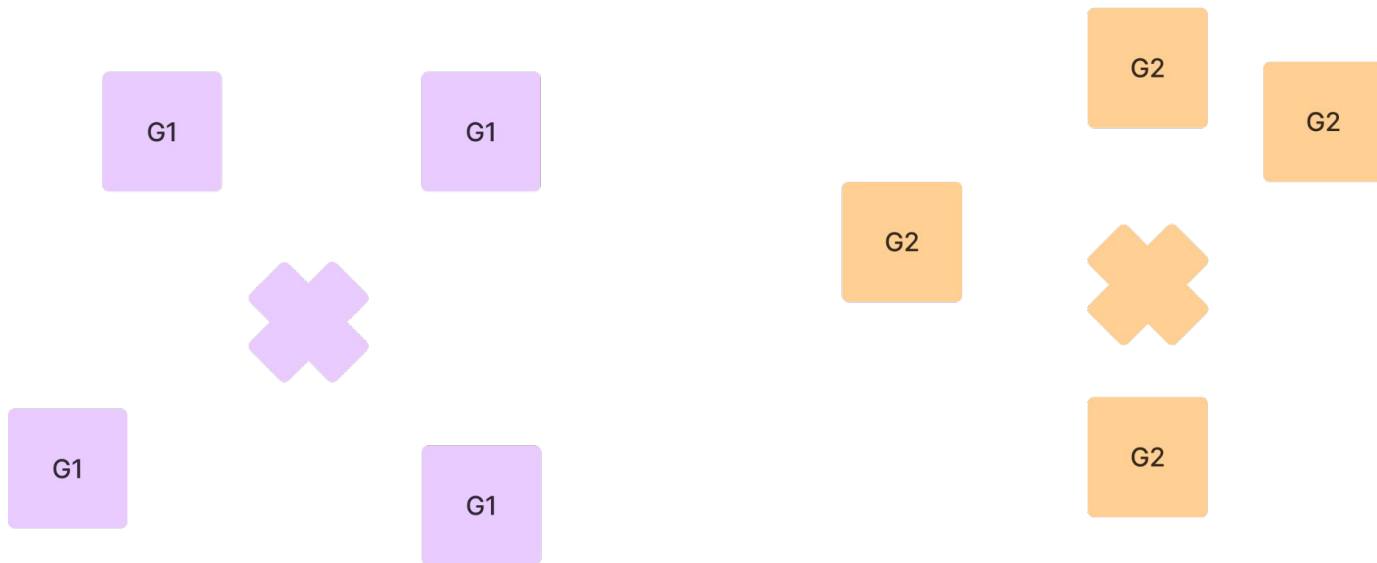
Getting predicted group

Centroid

Furniture image
from training dataset

G1

G1

G2

G2

G2

G1

G1

G2

Distance A

Distance B

Validation image

Nearer to centroid of Group 1,
so predicted label is Group 1
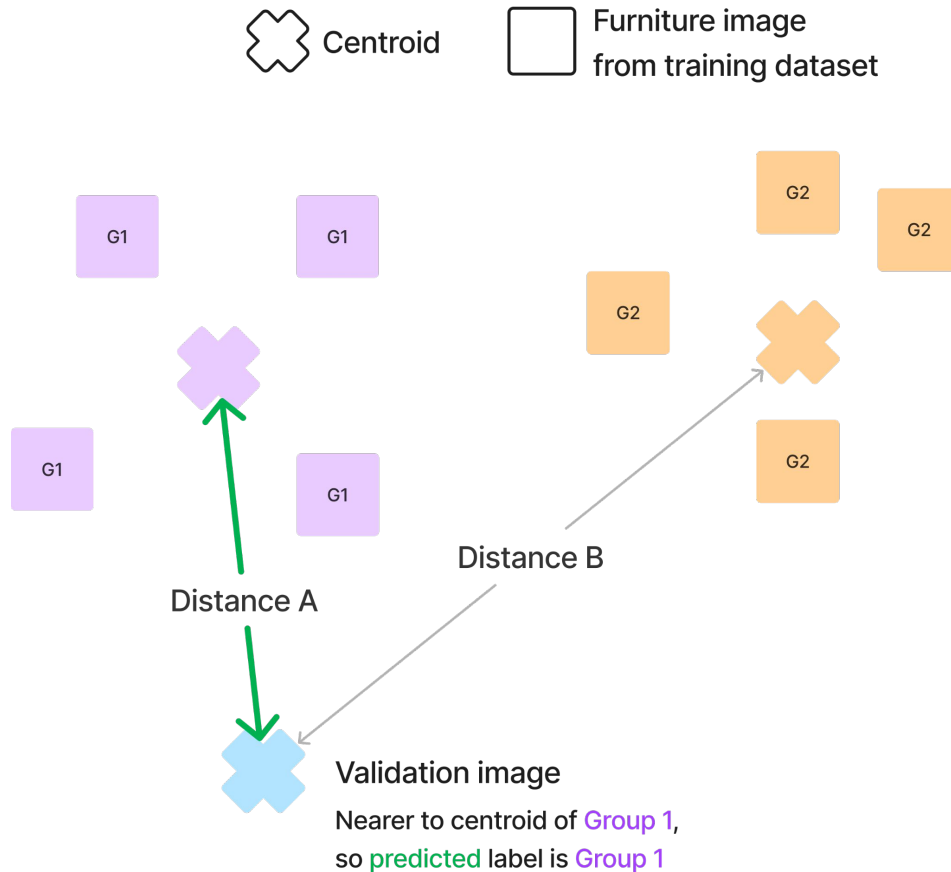
Precision score

- With the predicted and true group labels, we calculated the F1, precision and recall score

- Needed to choose a metric as a measure of success

To improve user experience:

Model must recommend only
relevant furniture items

Reduce number of false positives

Precision

Architectural
&
Hyperparameter Tuning

Reducing training set for hyperparameter tuning

I1

I2

I1, I2

I2, I1

I1, I2 are in the same group

I1, I2 | D1
I2, I1 | D1
I1, I2 | D2
I2, I1 | D2
I1, I2 | D3
I2, I1 | D3
. | .
. | .
. | .
I1, I2 | Dn
I2, I1 | Dn

Limit by n

Anchor, Positive   Negative
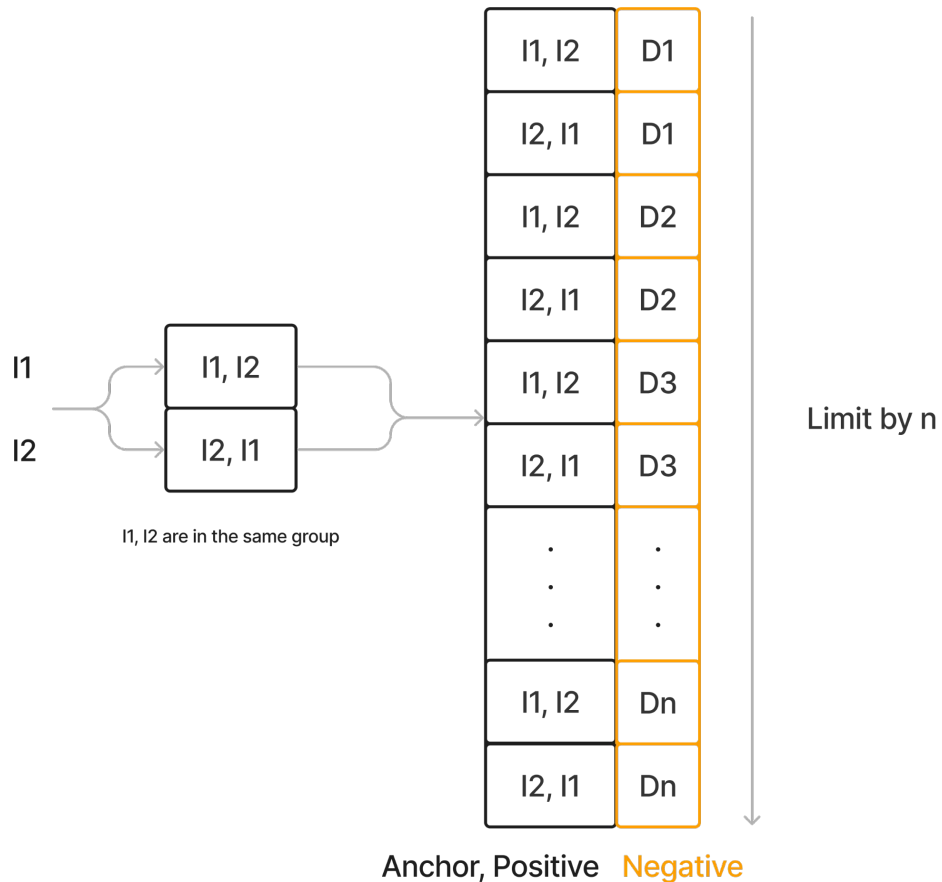
- Keep hyperparameter tuning relevant through a reduced training set

- Keep a high development velocity

- Same triplets generation script with a smaller **n**

Overview of hyperparameter tuning

Batch size: 32, 64, 128, 256, 512, 1024

Architecture: Baseline   B   C

Learning rate: 0.00001   0.000001   0.0000001

Triplet loss margin: 0.5   1.0   1.5   2.0   2.5   3.0
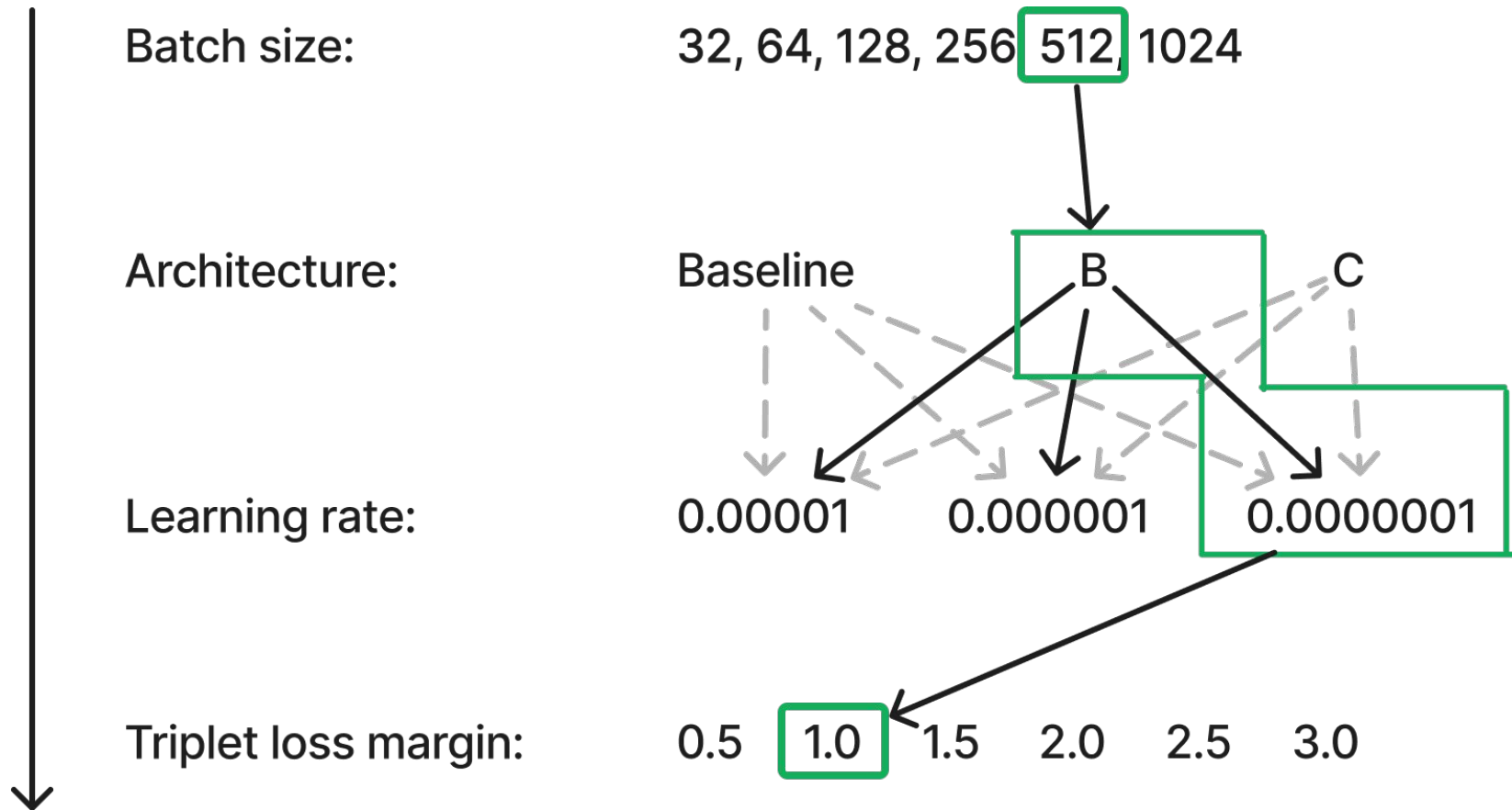
Choosing the optimal batch size

- Batch size determines our **training time** and computing resource consumption

- Shortest training time would give us <mark>quicker</mark> iteration cycles

Batch sizes tested:
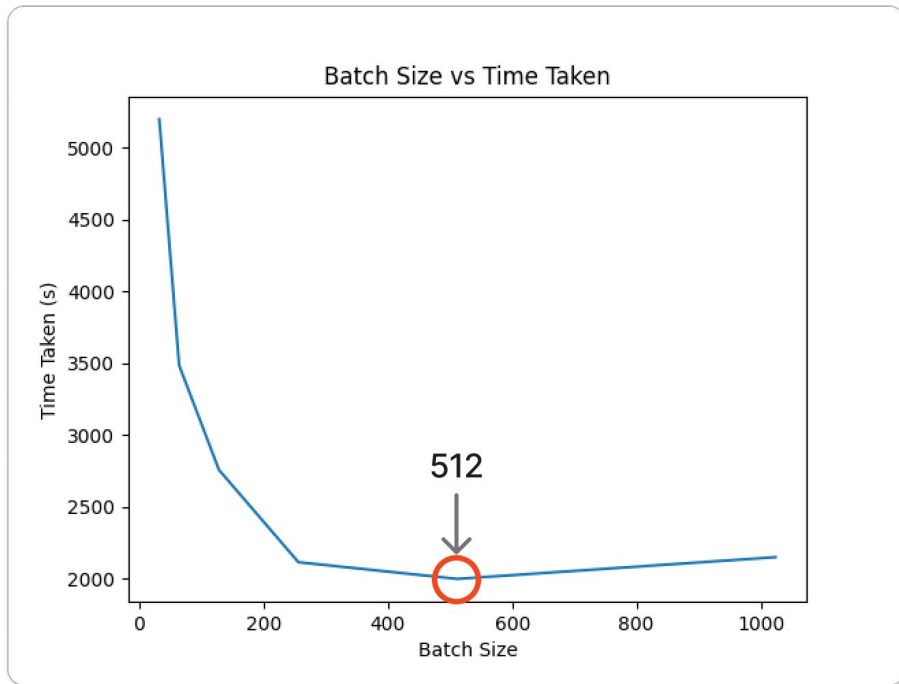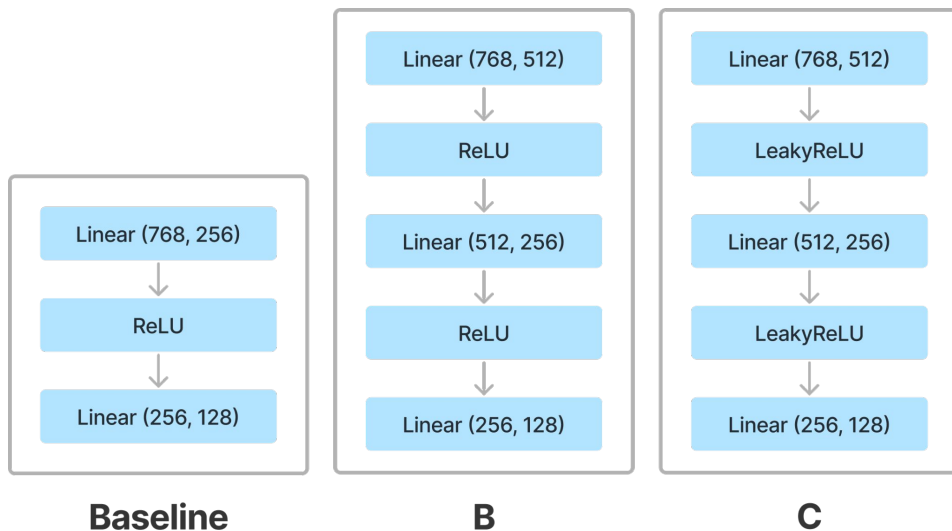
32, 64, 128, 256, 512, 1024

Optimal batch size:

512

Tuning architecture and learning rate

Baseline

Linear (768, 256)
ReLU
Linear (256, 128)

B

Linear (768, 512)
ReLU
Linear (512, 256)
ReLU
Linear (256, 128)

C

Linear (768, 512)
LeakyReLU
Linear (512, 256)
LeakyReLU
Linear (256, 128)

5 epochs
Triplet loss margin of 1.0
Batch size of 512
Constant learning rate

Learning rates:
0.00001, 0.000001, 0.0000001

- With over 700 groups, more layers might be needed
- Reduced representation dimensionality in smaller steps
- LeakyReLU may help in gaining more diverse feature representations

Architecture	Learning rate	Average Loss	Precision
Baseline	0.000001	0.089	0.52
B	0.0000001	0.714	**0.55**
C	0.00001	0.030	**0.55**

Tuning architecture and learning rate

| Architecture	Learning rate	Average Loss	Precision
Baseline	0.000001	0.089	0.52
B	0.0000001	**0.714**	0.55
C	0.00001	**0.030**	0.55
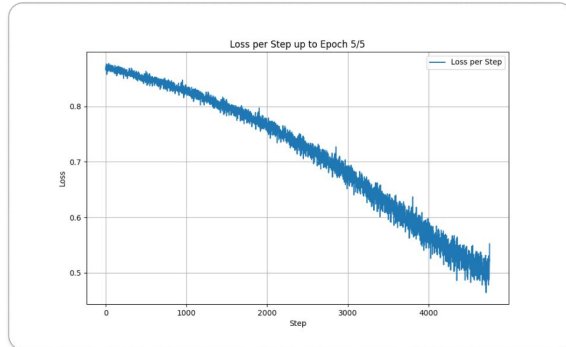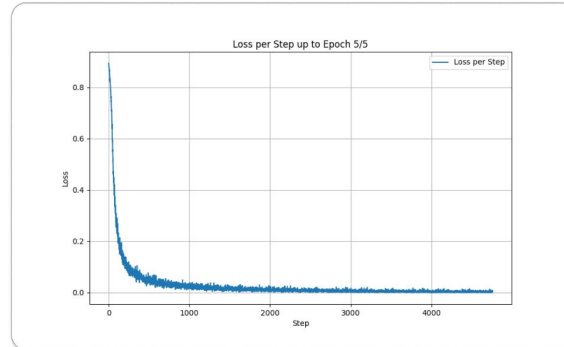
B has more room for improvement!

Triplet loss margin

An arbitrary range of triplet loss margin:
0.5, 1.0, 1.5, 2.0, 2.5, 3.0

	Average Loss	Precision
0.5	0.273	0.40
1.0	**0.714**	**0.55**
1.5	1.232	0.43
2.0	1.731	0.34
2.5	2.270	0.42
3.0	2.701	0.43

Final architecture & Hyperparameters

Linear (768, 512)

↓

ReLU

↓

Linear (512, 256)

↓

ReLU

↓

Linear (256, 128)
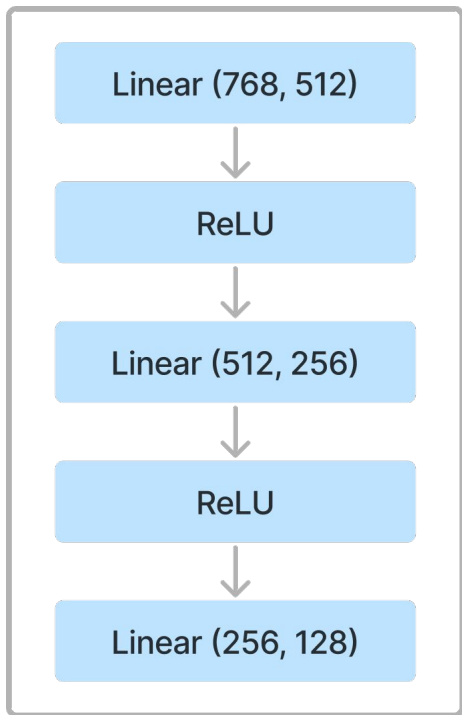
Batch size: 512

Triplet loss margin: 1.0

Constant Learning rate: 0.0000001

Training Results

Batch size: 512

Triplet loss margin: 1.0

Learning rate: 0.0000001

Constant learning rate

Epochs:
5, 10, 15, 20

	Precision
BLIP Only	0.46

Epochs	Average Loss	Precision
5	0.194	0.39
10	0.118	**0.43**
15	0.087	0.41
20	0.069	0.42

Improving the final model

With dropout

Epochs	Average Loss	Precision
5	0.2762	**0.43**
10	0.1865	0.36
15	0.08678	0.41
20	0.0688	0.40

With dropout (with lower LR)

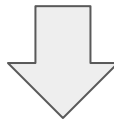| Epochs	Average Loss	Precision
5	0.409	**0.45**
10	0.280	0.37

10 epochs
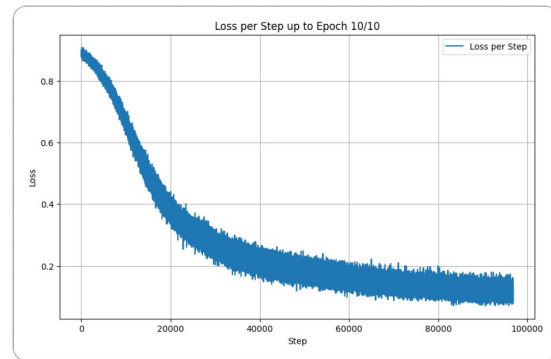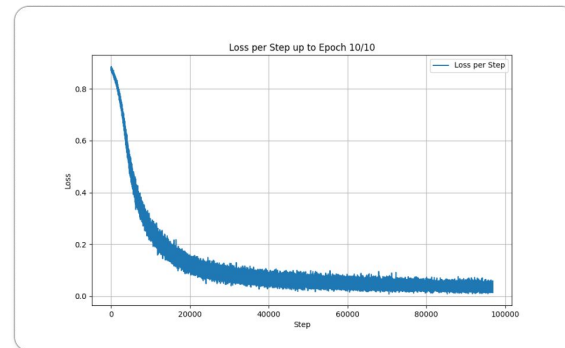
Model still converges and overfits early!

Potential improvements

Data preparation & preprocessing

Current

**Future
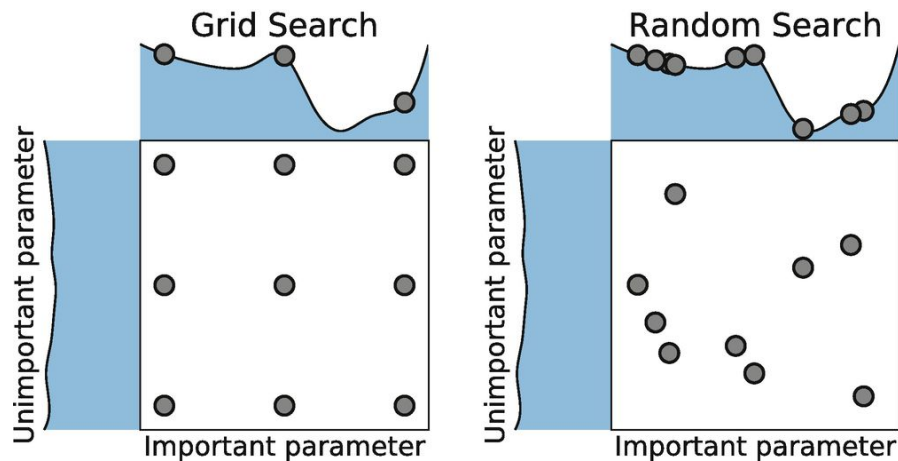Improvements**

Blur

Dropout

Multiple angles
(include existing images)

Image
augmentation

Tuning architecture & hyperparameters

Grid Search or Random Search

UI Demo