A Datasets & Preprocessing

We perform our experiments using three datasets from three different domains: MIMIC-III, ETTm1 and Taskonomy. The description of these three datasets are as follows:

Taskonomy Taskonomy is currently the largest multi-task dataset in the computer vision domain. It's a diverse dataset with data collected from the 3D scans of near 600 buildings. There are more than 4 million instances in the dataset. we follow [4, 1] to conduct the splits of train, validation, and test sets within $3,923,543$ training instances, $45,715$ validation instances and $48,362$ test instances. Additionally. All labels used in the Taskonomy dataset were normalized to have zero mean and unit standard deviation. The five vision tasks on the Taskonomy dataset are: *Semantic Segmentation*, *Depth Estimation*, *Surface Normal*, *Keypoint Detection*, and *Canny Edge Detection*. Though the number of combinations is limited for the Taskonomy dataset, due to over ten million parameters in the network and about four million images for training, it is still time-consuming to finish all MTL procedures, which take $5,472$ GPU hours in total.

ETTm1 ETTm1 dataset contains the data collected from an electricity transformer from a region of China. It contains 6 power load series and an oil temperature series that are recorded every 15 minutes range from July 2016 to July 2018. The 6 power load series are: HF (High useFul load), HL (High useLess load), MF (Middle useFul load), ML (Middle useLess load), LF (Low useFul load), LL (Low useLess load). We use the observations from all series as input, and treat a specific series output as a single forecasting task. We follow [6] to split the dataset into training, validation and test set in chronological order by the ratio of $6:2:2$. In total, the construction processes of the meta-learning data ($\mathcal{C}$, $\mathcal{Y}$) on Taskonomy cost 346 GPU hours.

MIMIC-III The Medical Information Mart for Intensive Care (MIMIC-III) dataset collects the de-identified electronic heal records (EHRs) of patients admitted to critical care units at a Boston-area hospital from 2001–2012. Regarding the MIMIC-III dataset, as noted by [2, 3], the medical scenarios usually did not hold enough data compared with machine learning applications in natural language or computer vision. Accordingly, MTL can play a key role in those scenarios to boost the performance of machine learning models. Therefore, we follow the few-shot setup in [2, 3] to construct the training data. In total, we evaluated the experiments on EHR records collected from patients. We divided the dataset into 1755 training instances, 500 validation instances, and 500 test instances. We follow [2, 3] to extract the data from MIMIC-III and construct the train, validation, and test sets correspondingly. [3] defined 12 task categories, each of which includes one or multiple specific tasks. For instance, there are 19 major icd codes, and each one corresponds to a binary prediction task. We conduct MTL procedures for 27 one-task combinations, 351 two-task combinations (for HOA-based methods), and $3,000$ randomly selected high-order combinations (as an approximation to $\mathcal{C}$). The selected tasks are provided in Table 1. A task that has a name with format icd $[\text{diagosename}]$ means this is a task for predicting the icd code diagose's presence. Other tasks' definitions are all follow [2, 3]. In total, the construction processes of the meta-learning data ($\mathcal{C}$, $\mathcal{Y}$) on MIMIC-III cost $4,138$ GPU hours.

We denote the processed datasets as `Taskonomy-5`, `ETTm1-7`, and `MIMIC-III-27`, respectively. We conduct all corresponding MTL procedures to collect the transferring gains on the validation set and record the final performance on the test set. During the MTL procedure for each task combination, we restart the training with five random seeds and report the average performance to eliminate the randomness in neural network training.

B MTL Backbone Models & Hyperparameters

In this section, we provide our experimental details for reproducibility, including the backbone models used for different datasets and their hyperparameter settings. All experiments are implemented in PyTorch and conducted on a single NVIDIA V100 16GB GPU.

For `MIMIC-III-27`, we use the model proposed in [3]. It leverages a two-layer GRU encoder for all tasks while each task has a task-specific fully-connected layer as the decoder. The input sequence length is set to 48. The number of GRU layers is 2, and the hidden size of the GRU layer is 512. The learning rate is set to 0.001. The dropout rate of GRU layers is 0.1. The batch size is 512. We leverage an early stopping strategy to train the model and the early stopping patience is 5. The

Datasets	Selected Tasks
Taskonomy-5	*Semantic Segmentation* (s) , *Depth Estimation* (d) , *Surface Normal* (n) , *Keypoint Detection* (k) , and *Edge Detection* (e)
ETTm1-7	High useFul load(HF) , High useLess load(HL) , Middle useFul load(MF) , Middle useLess load(ML) , High useFul load(LF) , High useLess load(LL) , Oil Temperature(OT)
MIMIC-III-27	*icd infection* (t_1) , *icd neoplasms* (t_2) , *icd endocrine* (t_3) , *icd blood* (t_4) , *icd mental* (t_5) , *icd nervous* (t_6) , *icd circulatory* (t_7) , *icd respiratory* (t_8) , *icd digestive* (t_9) , *icd genitourinary* (t_{10}) , *icd pregnancy* (t_{11}) , *icd skin* (t_{12}) , *icd musculoskeletal* (t_{13}) , *icd congenital* (t_{14}) , *icd ill defined* (t_{15}) , *icd injury* (t_{16}) , *dis 24h* (t_{17}) , *dis 48h* (t_{18}) , *mor 24h* (t_{19}) , *mor 48h* (t_{20}) , *los* (t_{21}) , *rea* (t_{22}) , *acu* (t_{23}) , *dnr 24h* (t_{24}) , *dnr 48h* (t_{25}) , *cmo 24h* (t_{26}) , *cmo 48h* (t_{27})

Table 1: Tasks for Taskonomy-5, MIMIC-III-5, MIMIC-III-27.

final results of different task combinations on MIMIC-III-27 dataset are the average results of 5 random seeds. We utilize cross-entropy loss for all tasks on MIMIC-III-27.Its code is available at: https://github.com/mmcdermott/comprehensive_MTL_EHR.

For Taskonomy-5, we follow the model used in [4]. It leverages a modified Xception network as the encoder, which has 36 convolutional layers forming the feature extraction base. As for task-specific decoders, it uses four transposed convolutional layers and four convolution layers. We follow the hyperparameters of Xception encoder used in [4] for Taskonomy-5 experiments, which modifies the original proposed Xception network for a low inference time. It replaces the max-pooling layers with 2×2 convolution layers with a stride of 2, the input image size is 256×256. The learning rate is set to 0.1, and the batch size is 16. For model training, we follow [4], which trains the model for 100 epochs and picks the model that has the best performance on the validation set. The final results of different task combinations on Taskonomy-5 dataset are the average results of 3 random seeds. We utilize cross-entropy loss for *Semantic Segmentation* and $L1$ loss for all other tasks for model optimization. Its code is available at: https://github.com/tstandley/taskgrouping.

For ETTm1-7, we follow the model used in [6]. It leverages an encoder-decoder model with decomposition capabilities and an approximation to attention based on Fourier transform. The input sequence length is set to 96, and the prediction length is set to 24. The number of encoder layers is 2, and the number of the decoder layer is 1. The number of attention heads in Autoformer's auto-correlation layers is 8. The dropout rate is 0.05. The hidden dimension of embedding in encoder layers and decoder layers is 512, and the dimension of the fully-connected layer is 2048. The learning rate is set to 0.0001 and the batch size is 32. We leverage an early stopping strategy to train the model and the early stopping patience is 10. The final results of different task combinations on ETTm1-7 dataset are the average results of 5 random seeds. We utilize $L2$ loss for all tasks on ETTm1-7. Its code is available at: https://github.com/thuml/Autoformer.

C Grouping Selection Algorithm

For Taskonomy-5 and ETTm1-7, we follow the brute-force search solution used in [4, 1]. Considering the overwhelming computational cost of applying the brute-force search on MIMIC-III-27, we utilize a beam search algorithm as an alternative. The beam search algorithm keeps track of *multiple best* search sequences with a specific beam size rather than *one* search sequence compared to greedy search algorithm. It provides a tradeoff between accuracy *v.s.* computational cost via its flexible choice of the beam size. The pseudo-code of the beam search algorithm is provided at Algorithm 1.

D Additional Experiments & Analyses

We provide additional experiments and experimental analyses in this section. Generally, we provide the case studies on three aspects: (1) We provide case studies on the task level gains, which includes the per-task gains on each dataset with different methods; (2) We provide further visualization results

Algorithm 1: Beam Search for Grouping Selection

Input: A set of task combinations $\mathcal{Y}$, all task combinations $\mathcal{Y}_a$, budget c_0, number of search beginning points β, remaining budget c_r, beam size γ. Initially, $c_r = 1$.

Function `beamSearch`($\mathcal{Y}_a$, c_0, β)**:**
- $runningList \leftarrow$ best β candidates in $\mathcal{Y}_a$;
- **while** $c_r \leq c_0$ **do**
  - **for** $List$ *in* $runningList$ **do**
    - $\mathcal{Y}$= `Filter`($\mathcal{Y}_a$, $List$);
    - **while** $\mathcal{Y}$ *is not empty* **do**
      - $\mathcal{X} \leftarrow \mathcal{Y}.pop$;
      - $newList \leftarrow List \cup \mathcal{X}$;
      - $runningList \leftarrow runningList \cup newList$
    - **end**
  - **end**
  - $runningList \leftarrow$ Best γ candidates in $runningList$;
  - $c_r = c_r + 1$;
- **end**
- $S \leftarrow \emptyset$;
- **for** $List \in runningList$ **do**
  - $S \leftarrow$ `Better`(S,$List$);
- **end**
- **return** S

Function `Filter`($\mathcal{Y}$,$List$)**:**
- Remove candidates that cannot improve $List$'s performance on any task.

Function `Better`(S_1, S_2)**:**
- $Score_1 \leftarrow$ total gain of S_1;
- $Score_2 \leftarrow$ total gain of S_2;
- **if** $Score_1 > Score_2$ **then**
  - **return** S_1
- **else**
  - **return** S_2

Output: Best group selection S

Table 2: Case studies on `Taskonomy-5`, we list the best task combinations identified by different methods.

Task	Methods	s	d	n	k	e	Estimated Gain	Valid Gain	Test Gain
s	HOA	✓	–	✓	–	–	+19.49%	+19.49%	+17.35%
	`MTG-Net`	✓	–	✓	–	–	+19.49%	+19.49%	+17.35%
	Oracle	✓	–	✓	–	–	–	+19.49%	+17.35%
d	HOA	–	✓	✓	–	–	+14.61%	+14.61%	+16.76%
	`MTG-Net`	–	✓	✓	–	–	+14.61%	+14.61%	+16.76%
	Oracle	–	✓	✓	–	–	–	+14.61%	+16.76%
n	HOA	–	–	✓	–	✓	+1.69%	+1.69%	+1.71%
	`MTG-Net`	–	✓	✓	–	✓	+1.80%	+1.80%	+1.82%
	Oracle	–	✓	✓	–	✓	–	+1.80%	+1.82%
k	HOA	–	–	✓	✓	–	+44.33%	+44.33%	+37.09%
	`MTG-Net`	–	–	✓	✓	–	+44.33%	+44.33%	+37.09%
	Oracle	–	–	✓	✓	–	–	+44.33%	+37.09%
e	HOA	–	–	✓	–	✓	+4.12%	+4.12%	+4.25%
	`MTG-Net`	–	✓	✓	–	✓	+11.04%	+11.04%	+12.14%
	Oracle	–	✓	✓	–	✓	–	+11.04%	+12.14%

Table 3: Case studies on ETTm1-7, we list the best task combinations identified by different methods.

Task	Methods	HF	HL	MF	ML	LF	LL	OT	Estimated Gain	Valid Gain	Test Gain
HF	HOA	✓	–	–	✓	–	–	–	+9.63%	+9.63%	+7.99%
	MTG-Net	✓	–	–	✓	✓	–	–	+19.20%	+19.20%	+16.43%
	Oracle	✓	–	–	✓	✓	–	–	-	+19.20%	+16.43%
HL	HOA	–	✓	–	✓	–	–	–	+4.84%	+4.84%	-2.41%
	MTG-Net	✓	✓	✓	✓	–	✓	✓	+7.57%	+7.57%	-1.15%
	Oracle	✓	✓	–	–	✓	–	–	-	+8.22%	-1.09%
MF	HOA	–	–	✓	–	✓	–	–	+12.75%	+12.75%	+20.68%
	MTG-Net	✓	✓	✓	–	✓	–	–	+16.08%	+16.08%	+16.53%
	Oracle	–	–	✓	✓	✓	–	✓	-	+24.60%	+23.70%
ML	HOA	–	✓	–	✓	–	–	–	+4.64%	+4.64%	+0.65%
	MTG-Net	✓	✓	✓	✓	–	✓	✓	+7.92%	+7.92%	+3.80%
	Oracle	–	–	✓	✓	✓	–	✓	-	+8.71%	+10.99%
LF	HOA	–	–	–	–	✓	–	–	0	0	0
	MTG-Net	–	–	–	✓	✓	✓	–	+6.61%	+6.61%	+5.57%
	Oracle	–	–	–	✓	✓	✓	–	-	+6.61%	+5.57%
LL	HOA	–	–	✓	–	–	✓	–	+0.99%	+0.99%	-10.36%
	MTG-Net	✓	✓	✓	✓	–	✓	-	+12.78%	+12.78%	+8.97%
	Oracle	✓	✓	✓	✓	–	✓	-	-	+12.78%	+8.97%
OT	HOA	✓	–	–	–	–	–	✓	+16.80%	+16.80%	+24.80%
	MTG-Net	✓	✓	✓	✓	–	✓	✓	+32.47%	+32.47%	+41.26%
	Oracle	✓	✓	✓	✓	–	✓	✓	-	+32.47%	+41.26%

on tasks' encoded embeddings; (3) We provide a comparison on estimation errors between active selection and random selection with different K.

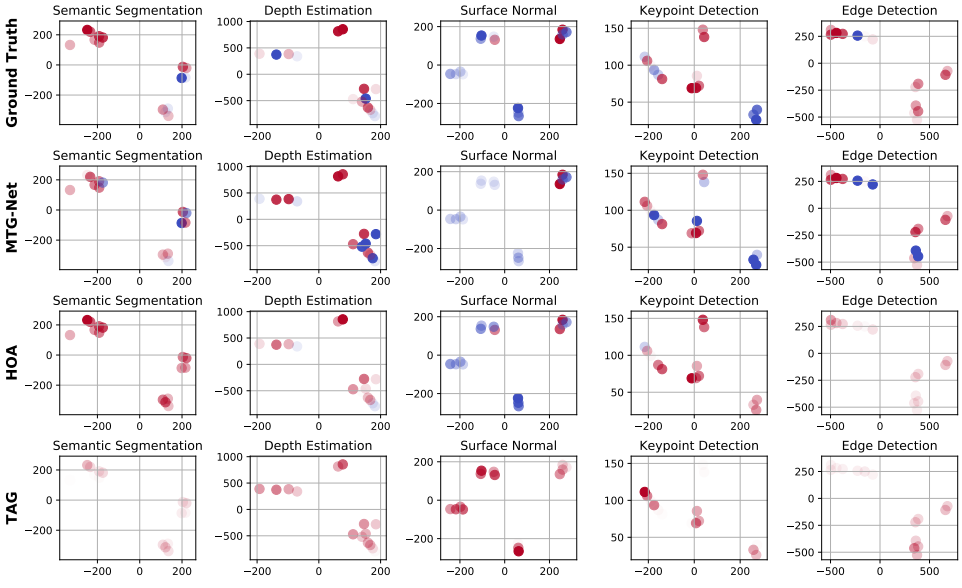
D.1 Case Study

For tasks on each dataset, we identify the task combination that provides the largest gain for it, and provide the best combination selection results for the all tasks on Taskonomy-5, ETTm1-7, and MIMIC-III-27, respectively.

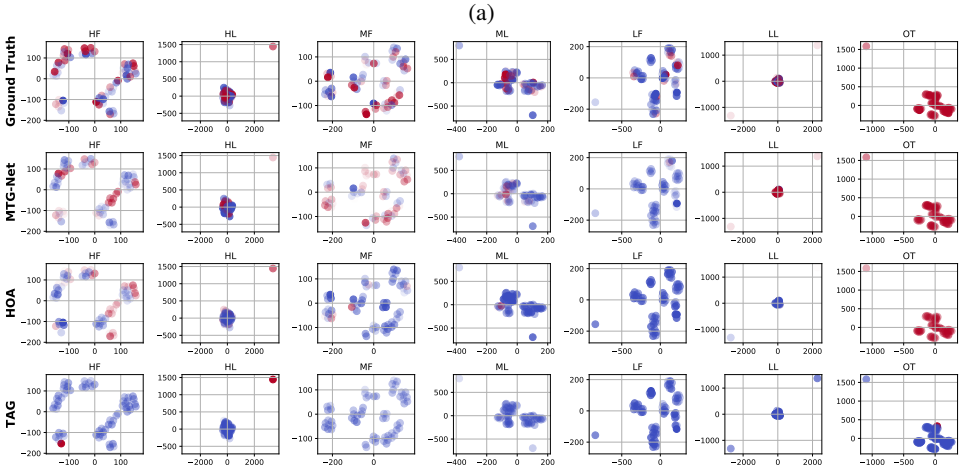
The experimental results can be seen through in Table 2, Table 3, and Table 4. For the combination selection results on Taskonmy-5, we can see the estimated gains of all tasks in this setting are the same as the ground truth. It depicts that such task combinations are sampled during training, and validates that MTG-Net has the ability to sample task combinations with large gains. Therefore, MTG-Net successfully identifies the optimal combination solutions for each task on Taskonmy-5. Furthermore, the results on MIMIC-III-27 show the generalization ability of MTG-Net, that is, it can predict and pick task combinations with large gains on unseen groups (t_3 and t_{24}). Thus, MTG-Net performs well on both settings for per-task best combination selection. Therefore, we think the final grouping performance advantages of MTG-Net are two-fold: (i) The active learning strategy encourages the model to sample task combinations with larger gains, which also makes it collect the ground truth gains of the sampled combinations. Thus, if the active learning strategy works, MTG-Net can collect the combinations with the largest gains through iterative sampling directly; (ii) Meta learning on task combinations allows MTG-Net to have the ability to generalize over the unseen task combinations. To sum up, the active learning strategy not only provides the 'fuel' for MTG-Net training, but also collects the combinations which possibly can be the optimal solutions for tasks. On the other hand, MTG-Net guides the sampling of active learning by predicting the performance of task combinations.

D.2 Further Visualization Results

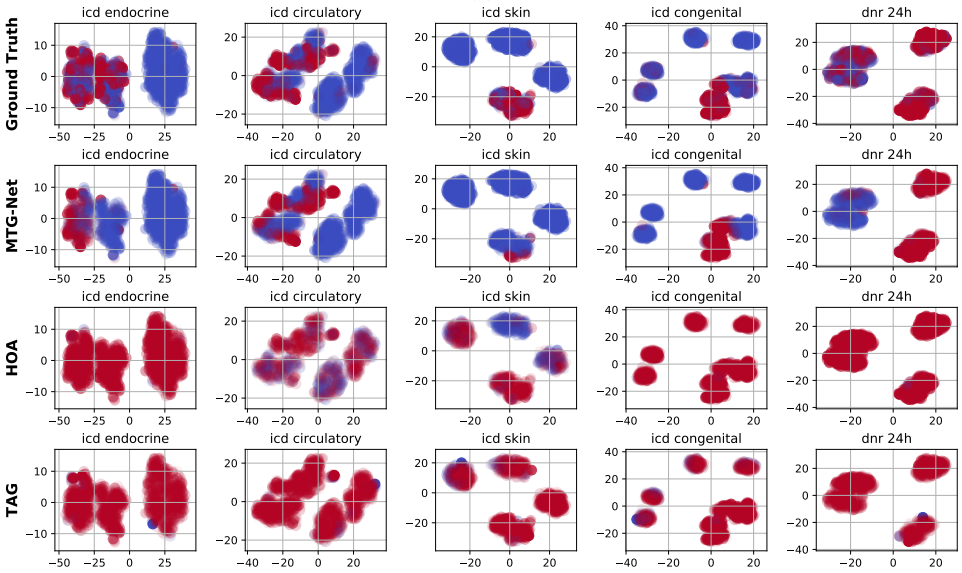
This section provides further visualization results and analyses on tasks' encoded embeddings, which are projected into a two dimensional space by t-SNE. The points' are colored by the ground truth value (red: positive; blue: negative; the deeper the color depth, the larger the gain (absolute


Ground Truth
MTG-Net
HOA
TAG
Semantic Segmentation
Depth Estimation
Surface Normal
Keypoint Detection
Edge Detection

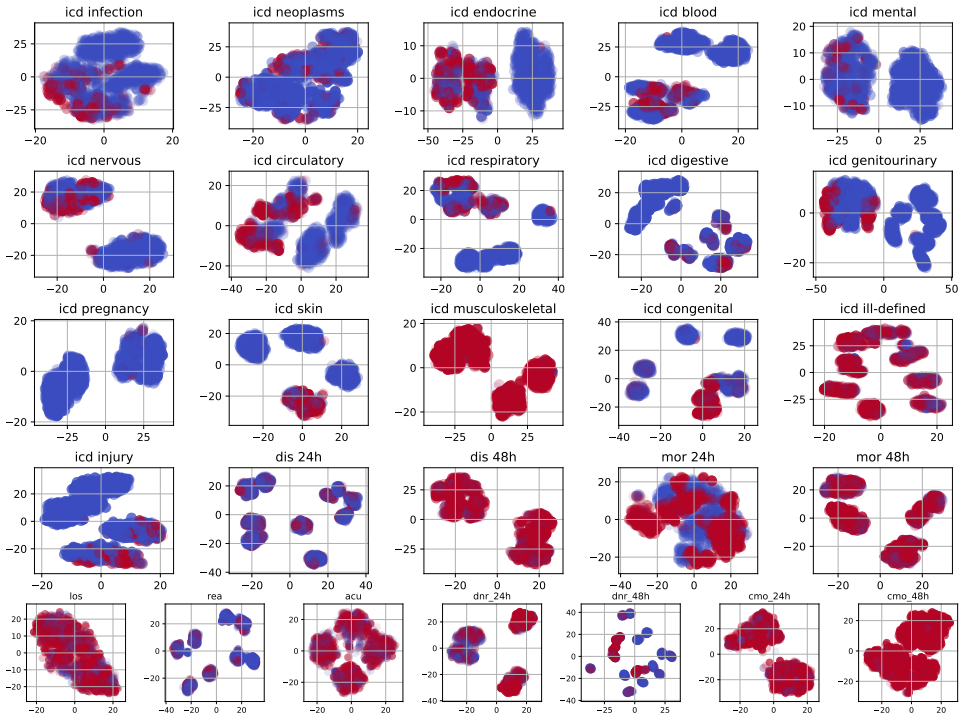
(a)


Ground Truth
MTG-Net
HOA
TAG
HF
HL
MF
ML
LF
LL
OT

(b)


Ground Truth
MTG-Net
HOA
TAG
icd endocrine
icd circulatory
icd skin
icd congenital
dnr 24h

(c)



(d)

Figure 1: We visualize the latent structures of transferring relationships discovered by `MTG-Net` in Figure 1a, Figure 1b, and Figure 1c. For each task, we aggregate its encoded embeddings for all relevant task combinations and project them into a two-dimensional space via t-SNE [5]. We color each combination by gain estimations that are produced by different approaches for this task. The combinations in the first row are colored by the ground-truth gains, while the combinations in the second to fourth row are colored by gain estimations produced by `MTG-Net`, `HOA`, and `TAG`, respectively. Figure 1a includes all 5 tasks in `Taskonomy-5`, Figure 1b includes all 7 tasks in `ETTm1-7` , and Figure 1c includes 5 selected tasks in `MIMIC-III-27`. Besides, Figure 1d shows the visualization of latent structures of transferring relationships that are colored by combinations' ground truth gains for all 27 tasks in `MIMIC-III-27`.

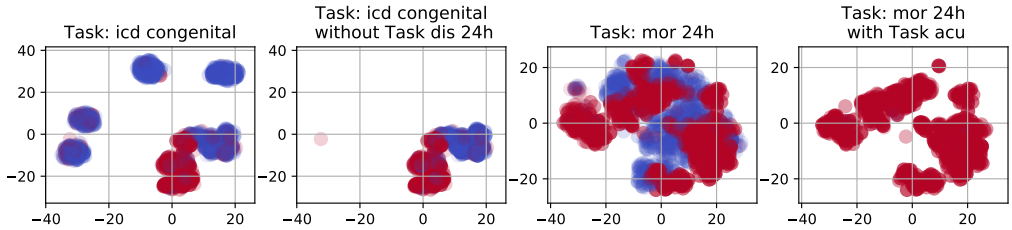


Figure 2: Two examples on pairwise task impact on the latent structures of transferring relationships discovered by `MTG-Net`. These diagrams illustrate the latent structures discovered by `MTG-Net` of two tasks: *icd congenital* and *mor 24h* in `MIMIC-III-27`. The first diagram illustrates *icd congenital*'s t-sne visualization results on all relevant task combinations, while the second diagram shows *icd congenital*'s t-sne visualization results on its relevant task combinations that exclude task *dis 24h*. The third diagram illustrates *mor 24h*'s t-sne visualization results on all relevant task combinations, while the second diagram shows *icd congenital*'s t-sne visualization results on its relevant task combinations that include task *dis 24h*.

Table 4: Case studies on MIMIC-III-27, we list the best task combinations identified by different methods.

Tasks	Methods	Selected Task Combinations	Estimated Gain	Valid Gain	Test Gain
t_3	HOA	t_3 t_{10}	+2.71%	+2.71%	-2.95%
	MTG-Net	t_1 t_2 t_3 t_4 t_7 t_8 t_{10} t_{11} t_{12} t_{13} t_{16} t_{20} t_{22} t_{24} t_{25} t_{26} t_{27}	+9.35%	+7.88%	+5.92%
	Oracle	t_1 t_2 t_3 t_7 t_8 t_{10} t_{11} t_{14} t_{15} t_{16} t_{22} t_{24} t_{26}	–	+8.35%	+6.65%
t_{11}	HOA	t_{11} t_{15}	+0.40%	+0.40%	+0.81%
	MTG-Net	t_{11}	0	0	0
	Oracle	t_{11} t_{15}	–	+0.40%	+0.81%
t_{16}	HOA	t_{16} t_{20}	+27.19%	+27.19%	+30.55%
	MTG-Net	t_1 t_2 t_3 t_4 t_7 t_8 t_{10} t_{11} t_{12} t_{13} t_{16} t_{20} t_{22} t_{24} t_{25} t_{26} t_{27}	+50.30%	+50.30%	+27.90%
	Oracle	t_5 t_6 t_9 t_{13} t_{14} t_{17} t_{18} t_{20} t_{21} t_{23} t_{25} t_{27}	–	+50.30%	+27.90%
t_{18}	HOA	t_{12} t_{18}	+2.22%	+2.22%	+0.70%
	MTG-Net	t_1 t_3 t_4 t_5 t_8 t_{10} t_{11} t_{12} t_{13} t_{15} t_{18} t_{21} t_{23} t_{24} t_{25} t_{26}	+10.10%	+5.73%	+2.58%
	Oracle	t_2 t_7 t_8 t_{10} t_{11} t_{15} t_{16} t_{18} t_{19} t_{20} t_{23} t_{26}	–	+6.12%	+8.37%
t_{19}	HOA	t_{19} t_{23}	+10.90%	+10.90%	+4.37%
	MTG-Net	t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18} t_{19} t_{20} t_{21} t_{22} t_{23} t_{24} t_{25} t_{26} t_{27}	+20.62%	+20.62%	+7.25%
	Oracle	t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18} t_{19} t_{20} t_{21} t_{22} t_{23} t_{24} t_{25} t_{26} t_{27}	–	+20.62%	+7.25%
t_{24}	HOA	t_{24} t_{27}	+9.00%	+9.00%	+8.49%
	MTG-Net	t_1 t_4 t_7 t_8 t_9 t_{12} t_{13} t_{14} t_{15} t_{16} t_{20} t_{21} t_{22} t_{23} t_{24}	+24.88%	+20.49%	+16.32%
	Oracle	t_1 t_2 t_3 t_4 t_6 t_8 t_{11} t_{12} t_{13} t_{15} t_{20} t_{21} t_{22} t_{23} t_{24} t_{27}	–	+26.10%	+21.74%

value)). In Figure 1a, Figure 1c and Figure 1b, we provide further visualization results on encoded embeddings by the predictions of different approaches. These figures show that MTG-Net learns the latent relationships between tasks more effectively compared with HOA and TAG. Besides, for a comprehensive view of the encoded representations of MIMIC-III-27's tasks, we provide the encoded task representations' t-SNE visualization results for all tasks in MIMIC-III-27 in Figure 1d.

Besides, we additionally provide experiments to show how pairwise impact on task encoded embedding's structures in Figure 2. The two diagrams on the left of Figure 2 illustrate the latent structures discovered by MTG-Net of task *icd congenital* in MIMIC-III-27. (Task *icd congenital* is a task to predict the presence of a congenital anomaly). The diagram on the left includes its t-sne visualization results on all relevant task combinations, while the diagram on the right shows its t-sne visualization results on its relevant task combinations exclude the *dis 24h*. However, as we can see, when we remove the task combinations that it's trained with *dis 24h* (*dis 24h* is used to predict whether the patients will be discharged within the next 24 hours), the *icd congenital*'s representation will no longer appear in some clusters that consist mainly of negative samples. It shows that there's a significant *negative transfer* from *dis 24h* to task *icd congenital*. Moreover, as it can be seen in the two diagrams on the right of Figure 2, when task *mor 24h* (whether the patient's recorded time of death is within the subsequent 24 hours) is trained with *acu* (if the patient will die, where will the patient stay at the end), its representation will no longer appear into a negative cluster, which suggests that there is a strong positive transfer between these two tasks. On the other hand, the task's

Dataset	Meta Model	Layers = 1	Layers = 2	Layers = 3	Layers = 4	Layers = 5
Taskonomy-5	MTG-Net (K=1)	+12.17%	+12.43%	+12.43%	+12.35%	+12.35%
Taskonomy-5	MTG-Net (K=2)	+17.90%	+18.34%	+18.34%	+17.90%	+18.34%
ETTm1-7	MTG-Net (K=1)	+10.52%	+10.61%	+10.61%	+10.57%	+10.52%
ETTm1-7	MTG-Net (K=3)	+14.44%	+14.52%	+14.50%	+14.44%	+14.50%
MIMIC-III-27	MTG-Net (K=1)	+6.25%	+6.28%	+6.29%	+6.29%	+6.31%
MIMIC-III-27	MTG-Net (K=13)	+8.63%	+8.73%	+8.74%	+8.76%	+8.78%

Table 5: Max-budget grouping performance of multiple MTG-Net models with different numbers of encoding layers.

Dataset	Meta Model	D = 8	D = 16	D = 32	D = 64	D = 128
Taskonomy-5	MTG-Net (K=1)	+12.35%	+12.43%	+12.43%	+12.43%	+12.35%
Taskonomy-5	MTG-Net (K=2)	+17.90%	+18.34%	+18.34%	+18.34%	+18.34%
ETTm1-7	MTG-Net (K=1)	+10.52%	+10.59%	+10.61%	+10.61%	+10.61%
ETTm1-7	MTG-Net (K=3)	+14.42%	+14.50%	+14.50%	+14.52%	+14.52%
MIMIC-III-27	MTG-Net (K=1)	+6.07%	+6.10%	+6.17%	+6.28%	+6.32%
MIMIC-III-27	MTG-Net (K=13)	+8.12%	+8.45%	+8.62%	+8.73%	+8.79%

Table 6: Max-budget grouping performance of multiple MTG-Net models with different numbers of hidden dimension.

encoded representation of MTG-Net successfully capture such task relationships and exhibits them in a structured latent space that can be seen from the visualization results above.

D.3 Hyper-parameter testing results for MTG-Net

This section supplements more hyper-parameter testing results for MTG-Net. We showed that MTG-Net is robust to different configurations of the number of encoding layers and the hidden dimension (D), which in this paper are set as 2 (layers) and 64 (D), respectively. Below we show the max-budget grouping performance of multiple MTG-Net models with different setups of these two hyper-parameters on MIMIC-III-27. The results can be seen in Table 5 and Table 6.

Moreover, from Table 7 we can see that MTG-Net can still obtain reasonable performance with other setups of α, and by significantly preferring task combinations with larger gains, MTG-Net is able to obtain slightly more improvements in terms of the final grouping performance. The rationale behind this observation is that the estimation errors on larger transferring gains can cause more direct error propagations to the grouping selection algorithm, thus paying more attention to these estimation errors helps most in the final grouping performance.

Besides, the hyper-parameter η controls the frequency of updating MTG-Net with actively selected meta samples, which allows a trade-off between efficiency and effectiveness for MTG-Net training. In this work, we adopt a dynamic strategy: 1) in the earliest round (K = 1), we set η as 1 to encourage more frequent MTG-Net updating per each active selection, 2) and in latter epochs (K > 1), we set η as N to further improve the efficiency by updating MTG-Net per N active selections. In the following table, we compare this dynamic strategy with two fixed strategies (η = 1 or η = N) to reveal the impact of η. From Table 8 we can see that our dynamic strategy for η can help MTG-Net with different Ks achieve competitive performance as setting η as 1 while ensuring that the number of MTG-Net updating steps is not an order of magnitude larger than that of setting η as N.

D.4 Detailed task combinations selected by MTG-Net at each step of active learning

Here we provide a few snapshots of the active learning procedure as a quick response in Table 9. From the table, we can observe diversified combinations being selected for different tasks at different

Dataset	Meta Model	α=0.01	α=0.1	α=1	α=10	α=25	α=100
Taskonomy-5	MTG-Net (K=1)	+11.35%	+11.50%	+11.95%	+12.43%	+12.43%	+12.43%
Taskonomy-5	MTG-Net (K=2)	+17. 75%	+17.90%	+18.14%	+18.34%	+18.34%	+18.34%
ETTm1-7	MTG-Net (K=1)	+10.11%	+10.15%	+10.52%	+10.61%	+10.61%	+10.61%
ETTm1-7	MTG-Net (K=3)	+14.30%	+14.37%	+14.46%	+14.52%	+14.52%	+14.52%
MIMIC-III-27	MTG-Net (K=1)	+5.86%	+6.07%	+6.15%	6.26%	+6.28%	+6.28%
MIMIC-III-27	MTG-Net (K=13)	+7.91%	+8.13%	+8.61%	+8.70%	+8.73%	+8.74%

Table 7: Max-budget grouping performance of MTG-Net with different α.

Meta Model	η	Updating Steps	Taskonomy-5	ETTm1-7	MIMIC-III-27
MTG-Net (K=1)	η = 1	$K * N \rightarrow N$	+12.47%	+10.65%	+6.29%
MTG-Net (K=1)	$\eta = N$	$K \rightarrow 1$	+11.27%	+9.97%	+5.97%
MTG-Net (K=1)	η = 1 if K =1 else N	$N + K - 1 \rightarrow N$	+12.43%	+10.61%	+6.28%
MTG-Net (K=$(N-1)/2$)	η = 1	$K * N \rightarrow N * (N-1)/2$	+18.34%	+14.55%	+8.75%
MTG-Net (K=$(N-1)/2$)	$\eta = N$	$K \rightarrow (N-1)/2$	+17.75%	+14.26%	+8.69%
MTG-Net (K=$(N-1)/2$)	η = 1 if K =1 else N	$N + K - 1 \rightarrow (N-1) * 3/2$	+18.34%	+14.52%	+8.73%

Table 8: Max-budget grouping performance with different setups of η and update steps.

rounds. More importantly, these results intuitively reveal that MTG-Net can calibrate its predictions after updating with the ground-truth gains.

D.5 How can MTG-Net benefits from the active learning strategy?

Figure 3 shows how the errors of gain estimations on unseen task combinations vary with K, which determines the scale of the meta-training set. We can observe that given the same size of $\mathcal{C}^{train}$, the active selection of meta-samples produces fewer estimation errors compared with the random selection. Combining Figure 3 with Figure 3 in the main text that illustrates random selection *v.s.* active selection in the main paper, we can observe that the performance of gain estimations highly correlates with the final grouping performance. Moreover, the active selection has a lower estimation error compared with random selection, and it performs better on the final grouping performance.

References

[1] Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. In *NeurIPS*, 2021.

[2] Matthew McDermott, Bret Nestor, Evan Kim, Wancong Zhang, Anna Goldenberg, Peter Szolovits, and Marzyeh Ghassemi. A comprehensive evaluation of multi-task learning and multi-task pre-training on ehr time-series data. *arXiv preprint arXiv:2007.10185*, 2020.

[3] Matthew McDermott, Bret Nestor, Evan Kim, Wancong Zhang, Anna Goldenberg, Peter Szolovits, and Marzyeh Ghassemi. A comprehensive ehr timeseries pre-training benchmark. In *CHIL*, 2021.

Dataset	Round K	Task j	Selected Combinations	Ground-truth Gain	Predicted Gain Before Selection	Predicted Gain After Selection
Taskonomy-5	1	s	[s,k]	6.3%	1.6%	3.2%
Taskonomy-5	2	s	[s,d,n,e]	7.2%	15.9%	9.1%
Taskonomy-5	1	n	[s,d,n]	-16.7%	-3.4%	-5.1%
Taskonomy-5	2	n	[n,e]	1.6%	-4.5%	0.2%
ETTm1-7	1	HF	[HF,MF,LL]	-5.5%	-2.3%	-6.5%
ETTm1-7	3	HF	[HF,ML,LF]	19.2%	5.4%	7.3%
ETTm1-7	1	LL	[LL,OT]	-3.5%	-5.5%	-2.8%
ETTm1-7	3	LL	[HF,HL,MF,LF,LL]	-2.0%	4.1%	-3.5%
MIMIC-III-27	1	t_1	[t_1,t_2,t_3,t_5,t_8,t_9,t_{10},t_{16}, t_{17},t_{19},t_{21},t_{23},t_{24},t_{25}]	-8.0%	-2.0%	-7.3%
MIMIC-III-27	7	t_1	[t_1,t_3,t_6,t_7,t_9,t_{10},t_{11}, t_{14},t_{16},t_{26},t_{27}]	4.4%	6.8%	5.7%
MIMIC-III-27	13	t_1	[[t_1,t_{12},t_{13},t_{14},t_{15},t_{17}, t_{18},t_{19},t_{24},t_{26}]	-7.7%	-9.3%	-8.4%
MIMIC-III-27	1	t_24	[t_4,t_8,t_9,t_{12},t_{13},t_{14},t_{17}, t_{20},t_{24}]	-3.2%	-36.1%	-5.1%
MIMIC-III-27	7	t_{24}	[t_8,t_{10},t_{11},t_{13},t_{15},t_{19},t_{21}, t_{22},t_{23},t_{24},t_{25}]	20.1%	20.7%	20.5%
MIMIC-III-27	13	t_24	[t_1,t_2,t_3,t_4,t_7,t_8,t_9,$t_1$1, $t_1$2,$t_1$6,$t_2$0,$t_2$2,$t_2$3,t_24]	21.1%	23.5%	22.3%

Table 9: Snapshots of the active learning procedure.

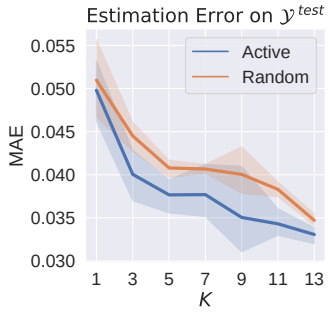


Figure 3: The estimation errors on $\mathcal{C}^{test}$ as K increases.

[4] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *ICML*, 2020.

[5] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008.

[6] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *NeurIPS*, 2021.