

---

**CRSLab**  
*Release 0.1.1*

**RUC AIBox**

**May 16, 2023**



## API REFERENCE

<b>1</b>	<b>crslab.quick_start package</b>	<b>3</b>
<b>2</b>	<b>crslab.config package</b>	<b>5</b>
<b>3</b>	<b>crslab.data package</b>	<b>7</b>
<b>4</b>	<b>crslab.evaluator package</b>	<b>23</b>
<b>5</b>	<b>crslab.model package</b>	<b>25</b>
<b>6</b>	<b>crslab.system package</b>	<b>59</b>
<b>7</b>	<b>Indices and tables</b>	<b>61</b>
	<b>Python Module Index</b>	<b>63</b>
	<b>Index</b>	<b>65</b>







---

**CHAPTER  
ONE**

---

**CRSLAB.QUICK\_START PACKAGE**

**1.1 Submodules**

**1.2 Module contents**



## CRSLAB.CONFIG PACKAGE

### 2.1 Submodules

```
class crslab.config.config.Config(config_file, gpu='-1', debug=False)  
Bases: object
```

Configurator module that load the defined parameters.

Load parameters and set log level.

#### Parameters

- **config\_file** (*str*) – path to the config file, which should be in yaml format. You can use default config provided in the [Github repo](#), or write it by yourself.
- **debug** (*bool, optional*) – whether to enable debug function during running. Defaults to False.

```
get(item, default=None)
```

Get value of corrsponding item in config

#### Parameters

- **item** (*str*) – key to query in config
- **default** (*optional*) – default value for item if not found in config. Defaults to None.

**Returns** value of corrsponding item in config

```
static load_yaml_configs(filename)
```

This function reads yaml file to build config dictionary

**Parameters** **filename** (*str*) – path to yaml config

**Returns** config

**Return type** dict

## 2.2 Module contents

Config module which loads parameters for the whole system.

`crslab.config.SAVE_PATH`  
where system to save.

**Type** str

`crslab.config.DATASET_PATH`  
where dataset to save.

**Type** str

`crslab.config.MODEL_PATH`  
where model related data to save.

**Type** str

`crslab.config.PRETRAIN_PATH`  
where pretrained model to save.

**Type** str

`crslab.config.EMBEDDING_PATH`  
where pretrained embedding to save, used for evaluate embedding related metrics.

**Type** str

## CRSLAB.DATA PACKAGE

### 3.1 Subpackages

#### 3.1.1 crslab.data.dataloader package

##### Submodules

`class crslab.data.dataloader.base.BaseDataLoader(opt, dataset)`

Bases: abc.ABC

Abstract class of dataloader

---

##### Notes

'scale' can be set in config to limit the size of dataset.

---

##### Parameters

- **opt** (`Config or dict`) – config for dataloader or the whole system.
- **dataset** – dataset

`conv_batchify(batch)`

batchify data for conversation after process.

**Parameters** `batch` (`list`) – processed batch dataset.

**Returns** batch data for the system to train conversation part.

`conv_interact(data)`

Process user input data for system to converse.

**Parameters** `data` – user input data.

**Returns** data for system in converse.

`conv_process_fn()`

Process whole data for conversation before batch\_fn.

**Returns** processed dataset. Defaults to return the same as `self.dataset`.

`get_conv_data(batch_size, shuffle=True)`

get\_data wrapper for conversation.

You can implement your own process\_fn in conv\_process\_fn, batch\_fn in conv\_batchify.

**Parameters**

- **batch\_size** (*int*) –
- **shuffle** (*bool, optional*) – Defaults to True.

**Yields** tuple or dict of *torch.Tensor* – batch data for conversation.

**get\_data** (*batch\_fn, batch\_size, shuffle=True, process\_fn=None*)

Collate batch data for system to fit

**Parameters**

- **batch\_fn** (*func*) – function to collate data
- **batch\_size** (*int*) –
- **shuffle** (*bool, optional*) – Defaults to True.
- **process\_fn** (*func, optional*) – function to process dataset before batchify. Defaults to None.

**Yields** tuple or dict of *torch.Tensor* – batch data for system to fit

**get\_policy\_data** (*batch\_size, shuffle=True*)

get\_data wrapper for policy.

You can implement your own process\_fn in self.policy\_process\_fn, batch\_fn in policy\_batchify.

**Parameters**

- **batch\_size** (*int*) –
- **shuffle** (*bool, optional*) – Defaults to True.

**Yields** tuple or dict of *torch.Tensor* – batch data for policy.

**get\_rec\_data** (*batch\_size, shuffle=True*)

get\_data wrapper for recommendation.

You can implement your own process\_fn in rec\_process\_fn, batch\_fn in rec\_batchify.

**Parameters**

- **batch\_size** (*int*) –
- **shuffle** (*bool, optional*) – Defaults to True.

**Yields** tuple or dict of *torch.Tensor* – batch data for recommendation.

**policy\_batchify** (*batch*)

batchify data for policy after process.

**Parameters** **batch** (*list*) – processed batch dataset.

**Returns** batch data for the system to train policy part.

**policy\_process\_fn** ()

Process whole data for policy before batch\_fn.

**Returns** processed dataset. Defaults to return the same as *self.dataset*.

**rec\_batchify** (*batch*)

batchify data for recommendation after process.

**Parameters** **batch** (*list*) – processed batch dataset.

**Returns** batch data for the system to train recommendation part.

---

```
rec_interact(data)
    process user input data for system to recommend.

    Parameters data – user input data.

    Returns data for system to recommend.

rec_process_fn()
    Process whole data for recommendation before batch_fn.

    Returns processed dataset. Defaults to return the same as self.dataset.

retain_recommender_targetReturns Recommender part of self.dataset.

class crslab.data.dataloader.kbrd.KBRDDataloader(opt, dataset, vocab)
Bases: crslab.data.dataloader.base.BaseDataLoader

Dataloader for model KBRD.
```

---

## Notes

You can set the following parameters in config:

- 'context\_truncate': the maximum length of context.
- 'response\_truncate': the maximum length of response.
- 'entity\_truncate': the maximum length of mentioned entities in context.

The following values must be specified in *vocab*:

- 'pad'
- 'start'
- 'end'
- 'pad\_entity'

the above values specify the id of needed special token.

---

## Parameters

- **opt** (*Config or dict*) – config for dataloader or the whole system.
- **dataset** – data for model.
- **vocab** (*dict*) – all kinds of useful size, idx and map between token and idx.

**conv\_batchify**(*batch*)
batchify data for conversation after process.

**Parameters** **batch** (*list*) – processed batch dataset.

**Returns** batch data for the system to train conversation part.

**conv\_process\_fn**(\**args*, \*\**kwargs*)
Process whole data for conversation before batch\_fn.

**Returns** processed dataset. Defaults to return the same as *self.dataset*.

```
policy_batchify (*args, **kwargs)
    batchify data for policy after process.

    Parameters batch (list) – processed batch dataset.

    Returns batch data for the system to train policy part.

rec_batchify (batch)
    batchify data for recommendation after process.

    Parameters batch (list) – processed batch dataset.

    Returns batch data for the system to train recommendation part.

rec_process_fn ()
    Process whole data for recommendation before batch_fn.

    Returns processed dataset. Defaults to return the same as self.dataset.
```

```
class crslab.data.dataloader.kgsf.KGSFDataLoader (opt, dataset, vocab)
Bases: crslab.data.dataloader.base.BaseDataLoader

Dataloader for model KGSF.
```

---

## Notes

You can set the following parameters in config:

- 'context\_truncate': the maximum length of context.
- 'response\_truncate': the maximum length of response.
- 'entity\_truncate': the maximum length of mentioned entities in context.
- 'word\_truncate': the maximum length of mentioned words in context.

The following values must be specified in *vocab*:

- 'pad'
- 'start'
- 'end'
- 'pad\_entity'
- 'pad\_word'

the above values specify the id of needed special token.

- 'n\_entity': the number of entities in the entity KG of dataset.
- 

## Parameters

- **opt** (*Config or dict*) – config for dataloader or the whole system.
- **dataset** – data for model.
- **vocab** (*dict*) – all kinds of useful size, idx and map between token and idx.

```
conv_batchify (batch)
    batchify data for conversation after process.

    Parameters batch (list) – processed batch dataset.

    Returns batch data for the system to train conversation part.
```

---

```
conv_process_fn(*args, **kwargs)
    Process whole data for conversation before batch_fn.

    Returns processed dataset. Defaults to return the same as self.dataset.

get_pretrain_data(batch_size, shuffle=True)

policy_batchify(*args, **kwargs)
    batchify data for policy after process.

    Parameters batch(list) – processed batch dataset.

    Returns batch data for the system to train policy part.

pretrain_batchify(batch)

rec_batchify(batch)
    batchify data for recommendation after process.

    Parameters batch(list) – processed batch dataset.

    Returns batch data for the system to train recommendation part.

rec_process_fn()
    Process whole data for recommendation before batch_fn.

    Returns processed dataset. Defaults to return the same as self.dataset.
```

**class** crslab.data.dataloader.redial.**ReDialDataLoader**(*opt*, *dataset*, *vocab*)  
Bases: *crslab.data.dataloader.base.BaseDataLoader*

Dataloader for model ReDial.

---

## Notes

You can set the following parameters in config:

- 'utterance\_truncate': the maximum length of a single utterance.
- 'conversation\_truncate': the maximum length of the whole conversation.

The following values must be specified in *vocab*:

- 'pad'
- 'start'
- 'end'
- 'unk'

the above values specify the id of needed special token.

- 'ind2tok': map from index to token.
- 'n\_entity': number of entities in the entity KG of dataset.
- 'vocab\_size': size of vocab.

---

## Parameters

- **opt** (*Config* or *dict*) – config for dataloader or the whole system.
- **dataset** – data for model.
- **vocab** (*dict*) – all kinds of useful size, idx and map between token and idx.

```
conv_batchify(batch)
    batchify data for conversation after process.

        Parameters batch (list) – processed batch dataset.

        Returns batch data for the system to train conversation part.

conv_process_fn()
    Process whole data for conversation before batch_fn.

        Returns processed dataset. Defaults to return the same as self.dataset.

policy_batchify(batch)
    batchify data for policy after process.

        Parameters batch (list) – processed batch dataset.

        Returns batch data for the system to train policy part.

rec_batchify(batch)
    batchify data for recommendation after process.

        Parameters batch (list) – processed batch dataset.

        Returns batch data for the system to train recommendation part.

rec_process_fn(*args, **kwargs)
    Process whole data for recommendation before batch_fn.

        Returns processed dataset. Defaults to return the same as self.dataset.
```

```
class crslab.data.dataloader.tgredial.TGReDialDataLoader(opt, dataset, vocab)
Bases: crslab.data.dataloader.base.BaseDataLoader

Dataloader for model TGReDial.
```

---

## Notes

You can set the following parameters in config:

- 'context\_truncate': the maximum length of context.
- 'response\_truncate': the maximum length of response.
- 'entity\_truncate': the maximum length of mentioned entities in context.
- 'word\_truncate': the maximum length of mentioned words in context.
- 'item\_truncate': the maximum length of mentioned items in context.

The following values must be specified in vocab:

- 'pad'
- 'start'
- 'end'
- 'unk'
- 'pad\_entity'
- 'pad\_word'

the above values specify the id of needed special token.

- 'ind2tok': map from index to token.

- 'tok2ind': map from token to index.
  - 'vocab\_size': size of vocab.
  - 'id2entity': map from index to entity.
  - 'n\_entity': number of entities in the entity KG of dataset.
  - 'sent\_split' (optional): token used to split sentence. Defaults to 'end'.
  - 'word\_split' (optional): token used to split word. Defaults to 'end'.
  - 'pad\_topic' (optional): token used to pad topic.
  - 'ind2topic' (optional): map from index to topic.
- 

### Parameters

- **opt** (`Config or dict`) – config for dataloader or the whole system.
- **dataset** – data for model.
- **vocab** (`dict`) – all kinds of useful size, idx and map between token and idx.

#### `conv_batchify(batch)`

batchify data for conversation after process.

**Parameters** `batch` (`list`) – processed batch dataset.

**Returns** batch data for the system to train conversation part.

#### `conv_interact(data)`

Process user input data for system to converse.

**Parameters** `data` – user input data.

**Returns** data for system in converse.

#### `policy_batchify(batch)`

batchify data for policy after process.

**Parameters** `batch` (`list`) – processed batch dataset.

**Returns** batch data for the system to train policy part.

#### `policy_process_fn(*args, **kwargs)`

Process whole data for policy before batch\_fn.

**Returns** processed dataset. Defaults to return the same as `self.dataset`.

#### `rec_batchify(batch)`

batchify data for recommendation after process.

**Parameters** `batch` (`list`) – processed batch dataset.

**Returns** batch data for the system to train recommendation part.

#### `rec_interact(data)`

process user input data for system to recommend.

**Parameters** `data` – user input data.

**Returns** data for system to recommend.

#### `rec_process_fn(*args, **kwargs)`

Process whole data for recommendation before batch\_fn.

**Returns** processed dataset. Defaults to return the same as *self.dataset*.

```
crslab.data.dataloader.utils.add_start_end_token_idx(vec: list, start_token_idx: Optional[int] = None, end_token_idx: Optional[int] = None)
```

Can choose to add start token in the beginning and end token in the end.

#### Parameters

- **vec** – source list composed of indexes.
- **start\_token\_idx** – index of start token.
- **end\_token\_idx** – index of end token.

**Returns** list added start or end token index.

**Return type** list

```
crslab.data.dataloader.utils.get_onehot(data_list, categories) → torch.Tensor
```

Transform lists of label into one-hot.

#### Parameters

- **data\_list** (*list of list of int*) – source data.
- **categories** (*int*) – #label class.

**Returns** one-hot labels.

**Return type** torch.Tensor

```
crslab.data.dataloader.utils.merge_utt(conversation, split_token_idx=None, keep_split_in_tail=False, final_token_idx=None)
```

merge utterances in one conversation.

#### Parameters

- **conversation** (*list of list of int*) – conversation consist of utterances consist of tokens.
- **split\_token\_idx** (*int*) – index of split token. Defaults to None.
- **keep\_split\_in\_tail** (*bool*) – split in tail or head. Defaults to False.
- **final\_token\_idx** (*int*) – index of final token. Defaults to None.

**Returns** tokens of all utterances in one list.

**Return type** list

```
crslab.data.dataloader.utils.merge_utt_replace(conversation, detect_token=None, replace_token=None, method='in')
```

```
crslab.data.dataloader.utils.padded_tensor(items: List[Union[List[int], torch.LongTensor]], pad_idx: int = 0, pad_tail: bool = True, max_len: Optional[int] = None) → torch.LongTensor
```

Create a padded matrix from an uneven list of lists.

Returns padded matrix.

Matrix is right-padded (filled to the right) by default, but can be left padded if the flag is set to True.

Matrix can also be placed on cuda automatically.

#### Parameters

- **items** (*list[iter[int]]*) – List of items
- **pad\_idx** (*int*) – the value to use for padding
- **pad\_tail** (*bool*) –
- **max\_len** (*int*) – if None, the max length is the maximum item length

**Returns** padded tensor.

**Return type** Tensor[int64]

`crslab.data.dataloader.utils.truncate(vec, max_length, truncate_tail=True)`  
truncate vec to make its length no more than max length.

**Parameters**

- **vec** (*list*) – source list.
- **max\_length** (*int*) –
- **truncate\_tail** (*bool, optional*) – Defaults to True.

**Returns** truncated vec.

**Return type** list

## Module contents

### 3.1.2 crslab.data.dataset package

#### Subpackages

##### crslab.data.dataset.durecdial package

#### Submodules

##### DuRecDial

---

#### References

Liu, Zeming, et al. “Towards Conversational Recommendation over Multi-Type Dialogs.” in ACL 2020.

---

```
class crslab.data.dataset.durecdial.DuRecDialDataset(opt, tokenize, restore=False, save=False)
Bases: crslab.data.dataset.base.BaseDataset

train_data
    train dataset.

valid_data
    valid dataset.

test_data
    test dataset.

vocab
```

```
{  
    'tok2ind': map from token to index,  
    'ind2tok': map from index to token,  
    'entity2id': map from entity to index,  
    'id2entity': map from index to entity,  
    'word2id': map from word to index,  
    'vocab_size': len(self.tok2ind),  
    'n_entity': max(self.entity2id.values()) + 1,  
    'n_word': max(self.word2id.values()) + 1,  
}
```

Type dict

---

## Notes

'unk' must be specified in 'special\_token\_idx' in resources.py.

---

## Parameters

- **opt** (`Config or dict`) – config for dataset or the whole system.
- **tokenize** (`str`) – how to tokenize dataset.
- **restore** (`bool`) – whether to restore saved dataset which has been processed. Defaults to False.
- **save** (`bool`) – whether to save dataset after processing. Defaults to False.

## Module contents

### crslab.data.dataset.gorecdial package

#### Submodules

#### GoRecDial

---

## References

Kang, Dongyeop, et al. “Recommendation as a Communication Game: Self-Supervised Bot-Play for Goal-oriented Dialogue.” in EMNLP 2019.

---

```
class crslab.data.dataset.gorecdial.GoRecDialDataset(opt, tokenize,  
                                                    restore=False,  
                                                    save=False)  
Bases: crslab.data.dataset.base.BaseDataset  
train_data  
    train dataset.  
valid_data  
    valid dataset.
```

**test\_data**  
test dataset.

**vocab**

```
{
    'tok2ind': map from token to index,
    'ind2tok': map from index to token,
    'entity2id': map from entity to index,
    'id2entity': map from index to entity,
    'word2id': map from word to index,
    'vocab_size': len(self.tok2ind),
    'n_entity': max(self.entity2id.values()) + 1,
    'n_word': max(self.word2id.values()) + 1,
}
```

Type dict

## Notes

'unk' must be specified in 'special\_token\_idx' in resources.py.

Specify tokenized resource and init base dataset.

## Parameters

- **opt** (Config or dict) – config for dataset or the whole system.
- **tokenize** (str) – how to tokenize dataset.
- **restore** (bool) – whether to restore saved dataset which has been processed. Defaults to False.
- **save** (bool) – whether to save dataset after processing. Defaults to False.

## Module contents

### crslab.data.dataset.inspired package

#### Submodules

#### Inspired

#### References

Hayati, Shirley Anugrah, et al. “INSPIRED: Toward Sociable Recommendation Dialog Systems.” in EMNLP 2020.

```
class crslab.data.dataset.inspired.inspired.InspiredDataset(opt, tokenize,
                                                               restore=False,
                                                               save=False)
```

Bases: crslab.data.dataset.base.BaseDataset

**train\_data**  
train dataset.

**valid\_data**  
valid dataset.

**test\_data**  
test dataset.

**vocab**

```
{  
    'tok2ind': map from token to index,  
    'ind2tok': map from index to token,  
    'entity2id': map from entity to index,  
    'id2entity': map from index to entity,  
    'word2id': map from word to index,  
    'vocab_size': len(self.tok2ind),  
    'n_entity': max(self.entity2id.values()) + 1,  
    'n_word': max(self.word2id.values()) + 1,  
}
```

Type dict

---

## Notes

'unk' must be specified in 'special\_token\_idx' in resources.py.

---

Specify tokenized resource and init base dataset.

### Parameters

- **opt** (Config or dict) – config for dataset or the whole system.
- **tokenize** (str) – how to tokenize dataset.
- **restore** (bool) – whether to restore saved dataset which has been processed. Defaults to False.
- **save** (bool) – whether to save dataset after processing. Defaults to False.

---

## Module contents

### crslab.data.dataset.opendialkg package

#### Submodules

##### OpenDialKG

---

#### References

Moon, Seungwhan, et al. “Opendifalkg: Explainable conversational reasoning with attention-based walks over knowledge graphs.” in ACL 2019.

---

```
class crslab.data.dataset.opendialkg.opendialkg.OpenDialKGDataset(opt, tok-  

enize, re-  

store=False,  

save=False)
```

Bases: crslab.data.dataset.base.BaseDataset

#### **train\_data**

train dataset.

#### **valid\_data**

valid dataset.

#### **test\_data**

test dataset.

#### **vocab**

```
{
    'tok2ind': map from token to index,
    'ind2tok': map from index to token,
    'entity2id': map from entity to index,
    'id2entity': map from index to entity,
    'word2id': map from word to index,
    'vocab_size': len(self.tok2ind),
    'n_entity': max(self.entity2id.values()) + 1,
    'n_word': max(self.word2id.values()) + 1,
}
```

Type dict

#### Notes

'unk' must be specified in 'special\_token\_idx' in resources.py.

Specify tokenized resource and init base dataset.

#### Parameters

- **opt** (`Config or dict`) – config for dataset or the whole system.
- **tokenize** (`str`) – how to tokenize dataset.
- **restore** (`bool`) – whether to restore saved dataset which has been processed. Defaults to False.
- **save** (`bool`) – whether to save dataset after processing. Defaults to False.

## Module contents

### crslab.data.dataset.redial package

#### Submodules

##### ReDial

---

#### References

Li, Raymond, et al. “Towards deep conversational recommendations.” in NeurIPS 2018.

---

```
class crslab.data.dataset.redial.ReDialDataset (opt, tokenize, restore=False,
                                                save=False)
```

Bases: crslab.data.dataset.base.BaseDataset

##### train\_data

train dataset.

##### valid\_data

valid dataset.

##### test\_data

test dataset.

##### vocab

```
{  
    'tok2ind': map from token to index,  
    'ind2tok': map from index to token,  
    'entity2id': map from entity to index,  
    'id2entity': map from index to entity,  
    'word2id': map from word to index,  
    'vocab_size': len(self.tok2ind),  
    'n_entity': max(self.entity2id.values()) + 1,  
    'n_word': max(self.word2id.values()) + 1,  
}
```

Type dict

---

#### Notes

'unk' must be specified in 'special\_token\_idx' in resources.py.

---

Specify tokenized resource and init base dataset.

#### Parameters

- **opt** (`Config or dict`) – config for dataset or the whole system.
- **tokenize** (`str`) – how to tokenize dataset.
- **restore** (`bool`) – whether to restore saved dataset which has been processed. Defaults to False.

- **save** (*bool*) – whether to save dataset after processing. Defaults to False.

## Module contents

### crslab.data.dataset.tgredial package

#### Submodules

##### TGReDial

#### References

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

```
class crslab.data.dataset.tgredial.tgredial.TGReDialDataset(opt, tokenize,  
                                         restore=False,  
                                         save=False)
```

Bases: crslab.data.dataset.base.BaseDataset

**train\_data**  
train dataset.

**valid\_data**  
valid dataset.

**test\_data**  
test dataset.

**vocab**

```
{
    'tok2ind': map from token to index,
    'ind2tok': map from index to token,
    'topic2ind': map from topic to index,
    'ind2topic': map from index to topic,
    'entity2id': map from entity to index,
    'id2entity': map from index to entity,
    'word2id': map from word to index,
    'vocab_size': len(self.tok2ind),
    'n_topic': len(self.topic2ind) + 1,
    'n_entity': max(self.entity2id.values()) + 1,
    'n_word': max(self.word2id.values()) + 1,
}
```

Type dict

#### Notes

'unk' and 'pad\_topic' must be specified in 'special\_token\_idx' in resources.py.

Specify tokenized resource and init base dataset.

### Parameters

- **opt** (`Config or dict`) – config for dataset or the whole system.
- **tokenize** (`str`) – how to tokenize dataset.
- **restore** (`bool`) – whether to restore saved dataset which has been processed. Defaults to False.
- **save** (`bool`) – whether to save dataset after processing. Defaults to False.

### Module contents

#### Submodules

#### Module contents

## 3.2 Module contents

Data module which reads, processes and batches data for the whole system

`crslab.data.dataset_register_table`  
record all supported dataset

Type dict

`crslab.data.dataset_language_map`  
record all dataset corresponding language

Type dict

`crslab.data.dataloader_register_table`  
record all model corresponding dataloader

Type dict

`crslab.data.get_dataloader(opt, dataset, vocab)` → `crslab.data.dataloader.base.BaseDataLoader`  
get dataloader to batchify dataset

#### Parameters

- **opt** (`Config or dict`) – config for dataloader or the whole system.
- **dataset** – processed raw data, no side data.
- **vocab** (`dict`) – all kinds of useful size, idx and map between token and idx.

**Returns** dataloader

`crslab.data.get_dataset(opt, tokenize, restore, save)` → `crslab.data.dataset.base.BaseDataset`  
get and process dataset

#### Parameters

- **opt** (`Config or dict`) – config for dataset or the whole system.
- **tokenize** (`str`) – how to tokenize the dataset.
- **restore** (`bool`) – whether to restore saved dataset which has been processed.
- **save** (`bool`) – whether to save dataset after processing.

**Returns** processed dataset

---

CHAPTER  
**FOUR**

---

## **CRSLAB.EVALUATOR PACKAGE**

### **4.1 Subpackages**

#### **4.1.1 crslab.evaluator.metrics package**

**Submodules**

**Module contents**

### **4.2 Submodules**

### **4.3 Module contents**



## CRSLAB.MODEL PACKAGE

### 5.1 Subpackages

#### 5.1.1 crslab.model.conversation package

##### Subpackages

###### crslab.model.conversation.gpt2 package

##### Submodules

###### GPT2

---

##### References

Radford, Alec, et al. “Language Models are Unsupervised Multitask Learners.”.

---

**class** crslab.model.conversation.gpt2.gpt2.**GPT2Model** (*opt, device, vocab, side\_data*)

Bases: *crslab.model.base.BaseModel*

**context\_truncate**

A integer indicating the length of dialogue context.

**response\_truncate**

A integer indicating the length of dialogue response.

**pad\_id**

A integer indicating the id of padding token.

##### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model()**

build model

**calculate\_loss** (*logit, labels*)

**Parameters**

- **preds** – torch.FloatTensor, shape=(bs, response\_truncate, vocab\_size)
- **labels** – torch.LongTensor, shape=(bs, response\_truncate)

**forward** (*batch, mode*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

**generate** (*context*)

**Parameters** **context** – torch.tensor, shape=(bs, context\_turncate)

**Returns** torch.tensor, shape=(bs, reponse\_turncate-1)

**Return type** generated\_response

**generate\_bs** (*context, beam=4*)

## Module contents

### crslab.model.conversation.transformer package

#### Submodules

#### Transformer

---

#### References

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

---

**class** crslab.model.conversation.transformer.transformer.**TransformerModel** (*opt, device, vocab, side\_data*)

Bases: *crslab.model.base.BaseModel*

**vocab\_size**

A integer indicating the vocabulary size.

**pad\_token\_idx**

A integer indicating the id of padding token.

**start\_token\_idx**

A integer indicating the id of start token.

**end\_token\_idx**

A integer indicating the id of end token.

**token\_emb\_dim**

A integer indicating the dimension of token embedding layer.

**pretrain\_embedding**

A string indicating the path of pretrained embedding.

**n\_word**

A integer indicating the number of words.

**n\_entity**

A integer indicating the number of entities.

**pad\_word\_idx**

A integer indicating the id of word padding.

**pad\_entity\_idx**

A integer indicating the id of entity padding.

**num\_bases**

A integer indicating the number of bases.

**kg\_emb\_dim**

A integer indicating the dimension of kg embedding.

**n\_heads**

A integer indicating the number of heads.

**n\_layers**

A integer indicating the number of layer.

**ffn\_size**

A integer indicating the size of ffn hidden.

**dropout**

A float indicating the dropout rate.

**attention\_dropout**

A integer indicating the dropout rate of attention layer.

**relu\_dropout**

A integer indicating the dropout rate of relu layer.

**learn\_positional\_embeddings**

A boolean indicating if we learn the positional embedding.

**embeddings\_scale**

A boolean indicating if we use the embeddings scale.

**reduction**

A boolean indicating if we use the reduction.

**n\_positions**

A integer indicating the number of position.

**longest\_label**

A integer indicating the longest length for response generation.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.

- **device** (`torch.device`) – A variable indicating which device to place the data and model.
- **vocab** (`dict`) – A dictionary record the vocabulary information.
- **side\_data** (`dict`) – A dictionary record the side data.

**\_starts** (`batch_size`)  
Return bsz start tokens.

**build\_model** ()  
build model

**forward** (`batch, mode`)  
Defines the computation performed at every call.  
Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### Module contents

#### 5.1.2 crslab.model.crs package

##### Subpackages

##### crslab.model.crs.kbrd package

##### Submodules

##### KBRD

---

##### References

Chen, Qibin, et al. “Towards Knowledge-Based Recommender Dialog System.” in EMNLP 2019.

---

**class** `crslab.model.crs.kbrd.KBRDModel` (`opt, device, vocab, side_data`)  
Bases: `crslab.model.base.BaseModel`

**vocab\_size**  
A integer indicating the vocabulary size.

**pad\_token\_idx**  
A integer indicating the id of padding token.

**start\_token\_idx**  
A integer indicating the id of start token.

**end\_token\_idx**  
A integer indicating the id of end token.

**token\_emb\_dim**  
A integer indicating the dimension of token embedding layer.

**pretrain\_embedding**  
A string indicating the path of pretrained embedding.

**n\_entity**  
A integer indicating the number of entities.

**n\_relation**  
A integer indicating the number of relation in KG.

**num\_bases**  
A integer indicating the number of bases.

**kg\_emb\_dim**  
A integer indicating the dimension of kg embedding.

**user\_emb\_dim**  
A integer indicating the dimension of user embedding.

**n\_heads**  
A integer indicating the number of heads.

**n\_layers**  
A integer indicating the number of layer.

**ffn\_size**  
A integer indicating the size of ffn hidden.

**dropout**  
A float indicating the dropout rate.

**attention\_dropout**  
A integer indicating the dropout rate of attention layer.

**relu\_dropout**  
A integer indicating the dropout rate of relu layer.

**learn\_positional\_embeddings**  
A boolean indicating if we learn the positional embedding.

**embeddings\_scale**  
A boolean indicating if we use the embeddings scale.

**reduction**  
A boolean indicating if we use the reduction.

**n\_positions**  
A integer indicating the number of position.

**longest\_label**  
A integer indicating the longest length for response generation.

**user\_proj\_dim**  
A integer indicating dim to project for user embedding.

### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.

- **device** (`torch.device`) – A variable indicating which device to place the data and model.
- **vocab** (`dict`) – A dictionary record the vocabulary information.
- **side\_data** (`dict`) – A dictionary record the side data.

**\_starts** (`batch_size`)  
Return bsz start tokens.

**build\_model** (\*args, \*\*kwargs)  
build model

**converse** (`batch, mode`)  
calculate loss and prediction of conversation for batch under certain mode

#### Parameters

- **batch** (`dict or tuple`) – batch data
- **mode** (`str, optional`) – train/valid/test.

**decode\_beam\_search** (`encoder_states, user_embedding, beam=4`)

**decode\_forced** (`encoder_states, user_embedding, resp`)

**decode\_greedy** (`encoder_states, user_embedding`)

**encode\_user** (`entity_lists, kg_embedding`)

**forward** (`batch, mode, stage`)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

**freeze\_parameters** ()

**recommend** (`batch, mode`)

calculate loss and prediction of recommendation for batch under certain mode

#### Parameters

- **batch** (`dict or tuple`) – batch data
- **mode** (`str, optional`) – train/valid/test.

## Module contents

### crslab.model.crs.kgsf package

#### Submodules

## KGSF

---

### References

Zhou, Kun, et al. "Improving Conversational Recommender Systems via Knowledge Graph based Semantic Fusion." in KDD 2020.

---

```
class crslab.model.crs.kgsf.KGSFModel (opt, device, vocab, side_data)
    Bases: crslab.model.base.BaseModel

vocab_size
    A integer indicating the vocabulary size.

pad_token_idx
    A integer indicating the id of padding token.

start_token_idx
    A integer indicating the id of start token.

end_token_idx
    A integer indicating the id of end token.

token_emb_dim
    A integer indicating the dimension of token embedding layer.

pretrain_embedding
    A string indicating the path of pretrained embedding.

n_word
    A integer indicating the number of words.

n_entity
    A integer indicating the number of entities.

pad_word_idx
    A integer indicating the id of word padding.

pad_entity_idx
    A integer indicating the id of entity padding.

num_bases
    A integer indicating the number of bases.

kg_emb_dim
    A integer indicating the dimension of kg embedding.

n_heads
    A integer indicating the number of heads.

n_layers
    A integer indicating the number of layer.

ffn_size
    A integer indicating the size of ffn hidden.

dropout
    A float indicating the dropout rate.

attention_dropout
    A integer indicating the dropout rate of attention layer.
```

**relu\_dropout**

A integer indicating the dropout rate of relu layer.

**learn\_positional\_embeddings**

A boolean indicating if we learn the positional embedding.

**embeddings\_scale**

A boolean indicating if we use the embeddings scale.

**reduction**

A boolean indicating if we use the reduction.

**n\_positions**

A integer indicating the number of position.

**response\_truncate = A integer indicating the longest length for response generation.**

**pretrained\_embedding**

A string indicating the path of pretrained embedding.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**\_starts** (*batch\_size*)

Return bsz start tokens.

**build\_model()**

build model

**converse** (*batch, mode*)

calculate loss and prediction of conversation for batch under certain mode

**Parameters**

- **batch** (*dict or tuple*) – batch data
- **mode** (*str, optional*) – train/valid/test.

**forward** (*batch, stage, mode*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

**freeze\_parameters()****pretrain\_infomax** (*batch*)

words: (*batch\_size, word\_length*) entity\_labels: (*batch\_size, n\_entity*)

---

```

recommend(batch, mode)
    context_entities: (batch_size, entity_length) context_words: (batch_size, word_length) movie:
    (batch_size)

class crslab.model.crs.kgsf.modules.GateLayer(input_dim)
Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

forward(input1, input2)
    Defines the computation performed at every call.

    Should be overridden by all subclasses.

```

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```

class crslab.model.crs.kgsf.modules.TransformerDecoderKG(n_heads, n_layers, em-
    bedding_size, ffn_size,
    vocabulary_size, em-
    bedding, dropout=0.0,
    attention_dropout=0.0,
    relu_dropout=0.0, em-
    beddings_scale=True,
    learn_positional_embeddings=False,
    padding_idx=None,
    n_positions=1024)

```

Bases: torch.nn.modules.module.Module

Transformer Decoder layer.

#### Parameters

- **n\_heads** (*int*) – the number of multihead attention heads.
- **n\_layers** (*int*) – number of transformer layers.
- **embedding\_size** (*int*) – the embedding sizes. Must be a multiple of n\_heads.
- **ffn\_size** (*int*) – the size of the hidden layer in the FFN
- **embedding** – an embedding matrix for the bottom layer of the transformer. If none, one is created for this encoder.
- **dropout** (*float*) – Dropout used around embeddings and before layer layer normalizations. This is used in Vaswani 2017 and works well on large datasets.
- **attention\_dropout** (*float*) – Dropout performed after the multhead attention softmax. This is not used in Vaswani 2017.
- **relu\_dropout** (*float*) – Dropout used after the ReLU in the FFN. Not used in Vaswani 2017, but used in Tensor2Tensor.
- **padding\_idx** (*int*) – Reserved padding index in the embeddings matrix.
- **learn\_positional\_embeddings** (*bool*) – If off, sinusoidal embeddings are used. If on, position embeddings are learned from scratch.
- **embeddings\_scale** (*bool*) – Scale embeddings relative to their dimensionality. Found useful in fairseq.

- **n\_positions** (*int*) – Size of the position embeddings matrix.

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward**(*input*, *encoder\_state*, *kg\_encoder\_output*, *kg\_encoder\_mask*, *db\_encoder\_output*, *db\_encoder\_mask*, *incr\_state=None*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.crs.kgsf.modules.TransformerDecoderLayerKG(n_heads,      em-
                                                               bedding_size,
                                                               ffn_size,      atten-
                                                               tion_dropout=0.0,
                                                               relu_dropout=0.0,
                                                               dropout=0.0)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward**(*x*, *encoder\_output*, *encoder\_mask*, *kg\_encoder\_output*, *kg\_encoder\_mask*, *db\_encoder\_output*, *db\_encoder\_mask*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### crslab.model.crs.redial package

#### Submodules

```
class crslab.model.crs.redial.modules.HRNN(utterance_encoder_hidden_size,      di-
                                              dialog_encoder_hidden_size,      dia-
                                              log_encoder_num_layers,      pad_token_idx,
                                              embedding=None,      use_dropout=False,
                                              dropout=0.3)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward**(*context*, *utterance\_lengths*, *dialog\_lengths*)

#### Parameters

- **context** – (batch\_size, max\_context\_length, max\_utterance\_length)
- **utterance\_lengths** – (batch\_size, max\_context\_length)

- **dialog\_lengths** – (batch\_size)

**Return context\_state** (batch\_size, context\_encoder\_hidden\_size)

**get\_utterance\_encoding** (context, utterance\_lengths)

**Parameters**

- **context** – (batch\_size, max\_conversation\_length, max\_utterance\_length)
- **utterance\_lengths** – (batch\_size, max\_conversation\_length)

**Return utterance\_encoding** (batch\_size, max\_conversation\_length, 2 \* utterance\_encoder\_hidden\_size)

```
class crslab.model.crs.redial.modules.SwitchingDecoder(hidden_size, context_size,
                                                       num_layers, vocab_size,
                                                       embedding, pad_token_idx)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (request, request\_lengths, context\_state)

**Parameters**

- **request** – (batch\_size, max\_utterance\_length)
- **request\_lengths** – (batch\_size)
- **context\_state** – (batch\_size, context\_encoder\_hidden\_size)

**Return log\_probabilities** (batch\_size, max\_utterance\_length, vocab\_size + 1)

## ReDial\_Conv

---

### References

Li, Raymond, et al. “Towards deep conversational recommendations.” in NeurIPS.

---

```
class crslab.model.crs.redial.redial_conv.ReDialConvModel(opt, device, vocab,
                                                          side_data)
```

Bases: *crslab.model.base.BaseModel*

**vocab\_size**  
A integer indicating the vocabulary size.

**pad\_token\_idx**  
A integer indicating the id of padding token.

**start\_token\_idx**  
A integer indicating the id of start token.

**end\_token\_idx**  
A integer indicating the id of end token.

**unk\_token\_idx**  
A integer indicating the id of unk token.

**pretrained\_embedding**  
A string indicating the path of pretrained embedding.

**embedding\_dim**

A integer indicating the dimension of item embedding.

**utterance\_encoder\_hidden\_size**

A integer indicating the size of hidden size in utterance encoder.

**dialog\_encoder\_hidden\_size**

A integer indicating the size of hidden size in dialog encoder.

**dialog\_encoder\_num\_layers**

A integer indicating the number of layers in dialog encoder.

**use\_dropout**

A boolean indicating if we use the dropout.

**dropout**

A float indicating the dropout rate.

**decoder\_hidden\_size**

A integer indicating the size of hidden size in decoder.

**decoder\_num\_layers**

A integer indicating the number of layer in decoder.

**decoder\_embedding\_dim**

A integer indicating the dimension of embedding in decoder.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model()**

build model

**forward(batch, mode)****Parameters batch –**

```
{  
    'context': (batch_size, max_context_length, max_utterance_  
    ↴length),  
    'context_lengths': (batch_size),  
    'utterance_lengths': (batch_size, max_context_length),  
    'request': (batch_size, max_utterance_length),  
    'request_lengths': (batch_size),  
    'response': (batch_size, max_utterance_length)  
}
```

## ReDial\_Rec

---

### References

Li, Raymond, et al. “Towards deep conversational recommendations.” in NeurIPS.

---

```
class crslab.model.crs.redial.redial_rec.ReDialRecModel (opt, device, vocab,  
                                                  side_data)
```

Bases: *crslab.model.base.BaseModel*

#### **n\_entity**

A integer indicating the number of entities.

#### **layer\_sizes**

A integer indicating the size of layer in autorec.

#### **pad\_entity\_idx**

A integer indicating the id of entity padding.

#### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

#### **build\_model()**

build model

#### **forward(*batch*, *mode*)**

##### Parameters

- **batch** –

```
{  
    'context_entities': (batch_size, n_entity),  
    'item': (batch_size)  
}
```

- **mode** (*str*) –

### Module contents

## crslab.model.crs.tgredial package

### Submodules

#### TGReDial\_Conv

---

### References

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

---

**class** `crslab.model.crs.tgredial.tg_conv.TGConvModel` (*opt, device, vocab, side\_data*)  
Bases: `crslab.model.base.BaseModel`

**context\_truncate**

A integer indicating the length of dialogue context.

**response\_truncate**

A integer indicating the length of dialogue response.

**pad\_id**

A integer indicating the id of padding token.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (`torch.device`) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model()**  
build model

**calculate\_loss** (*logit, labels*)

**Parameters**

- **preds** – `torch.FloatTensor`, shape=(bs, response\_truncate, vocab\_size)
- **labels** – `torch.LongTensor`, shape=(bs, response\_truncate)

**forward** (*batch, mode*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

**generate** (*context*)

**Parameters** **context** – `torch.tensor`, shape=(bs, context\_turncate)

**Returns** `torch.tensor`, shape=(bs, reponse\_turncate-1)

**Return type** generated\_response

**generate\_bs** (*context, beam=4*)

## TGReDial\_Policy

---

### References

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

---

```
class crslab.model.crs.tgredial.tg_policy.TGPolicyModel (opt, device, vocab,
                                                               side_data)
```

Bases: *crslab.model.base.BaseModel*

#### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model** (\*args, \*\*kwargs)  
build model

**forward** (batch, mode)  
Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## TGReDial\_Rec

---

### References

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

---

```
class crslab.model.crs.tgredial.tg_rec.TGRecModel (opt, device, vocab, side_data)
```

Bases: *crslab.model.base.BaseModel*

#### hidden\_dropout\_prob

A float indicating the dropout rate to dropout hidden state in SASRec.

#### initializer\_range

A float indicating the range of parameters initialization in SASRec.

#### hidden\_size

A integer indicating the size of hidden state in SASRec.

#### max\_seq\_length

A integer indicating the max interaction history length.

#### item\_size

A integer indicating the number of items.

**num\_attention\_heads**

A integer indicating the head number in SASRec.

**attention\_probs\_dropout\_prob**

A float indicating the dropout rate in attention layers.

**hidden\_act**

A string indicating the activation function type in SASRec.

**num\_hidden\_layers**

A integer indicating the number of hidden layers in SASRec.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model()**

build model

**forward(*batch, mode*)**

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

**Module contents****Module contents****5.1.3 crslab.model.policy package****Subpackages****crslab.model.policy.conv\_bert package****Submodules****Conv\_BERT**

---

**References**

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

---

---

```
class crslab.model.policy.conv_bert.conv_bert.ConvBERTModel (opt, device, vocab,  
 side_data)
```

Bases: *crslab.model.base.BaseModel*

**topic\_class\_num**  
the number of topic.

#### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model** (\**args*, \*\**kwargs*)  
build model

**forward** (*batch*, *mode*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the *Module* instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### crslab.model.policy.mgcf package

#### Submodules

##### MGCG

---

#### References

Liu, Zeming, et al. “Towards Conversational Recommendation over Multi-Type Dialogs.” in ACL 2020.

---

```
class crslab.model.policy.mgcf.mgcf.MGCGModel (opt, device, vocab, side_data)
```

Bases: *crslab.model.base.BaseModel*

**topic\_class\_num**  
A integer indicating the number of topic.

**vocab\_size**  
A integer indicating the size of vocabulary.

**embedding\_dim**  
A integer indicating the dimension of embedding layer.

**hidden\_size**

A integer indicating the size of hidden state.

**num\_layers**

A integer indicating the number of layers in GRU.

**dropout\_hidden**

A float indicating the dropout rate of hidden state.

**n\_sent**

A integer indicating sequence length in user profile.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model** (\*args, \*\*kwargs)  
build model

**forward**(batch, mode)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

**get\_length**(input)

**Module contents****crslab.model.policy.pmi package****Submodules****PMI**

**class** crslab.model.policy.pmi.pmi.**PMIModel**(*opt, device, vocab, side\_data*)  
Bases: *crslab.model.base.BaseModel*

**topic\_class\_num**

A integer indicating the number of topic.

**pad\_topic**

A integer indicating the id of topic padding.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.

- **device** (`torch.device`) – A variable indicating which device to place the data and model.
- **vocab** (`dict`) – A dictionary record the vocabulary information.
- **side\_data** (`dict`) – A dictionary record the side data.

**build\_model** (\*args, \*\*kwargs)  
build model

**forward** (batch, mode)  
Defines the computation performed at every call.  
Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### crslab.model.policy.profile\_bert package

#### Submodules

#### Profile\_BERT

---

#### References

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

---

**class** crslab.model.policy.profile\_bert.profile\_bert.**ProfileBERTModel** (opt,  
device,  
vocab,  
side\_data)

Bases: `crslab.model.base.BaseModel`

**topic\_class\_num**

A integer indicating the number of topic.

**n\_sent**

A integer indicating sequence length in user profile.

#### Parameters

- **opt** (`dict`) – A dictionary record the hyper parameters.
- **device** (`torch.device`) – A variable indicating which device to place the data and model.
- **vocab** (`dict`) – A dictionary record the vocabulary information.
- **side\_data** (`dict`) – A dictionary record the side data.

```
build_model (*args, **kwargs)
    build model

forward (batch, mode)
    Defines the computation performed at every call.
    Should be overridden by all subclasses.
```

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### crslab.model.policy.topic\_bert package

#### Submodules

##### Topic\_BERT

---

#### References

Zhou, Kun, et al. “Towards Topic-Guided Conversational Recommender System.” in COLING 2020.

---

```
class crslab.model.policy.topic_bert.topic_bert.TopicBERTModel (opt, device, vocab, side_data)
Bases: crslab.model.base.BaseModel
```

**topic\_class\_num**  
A integer indicating the number of topic.

#### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

```
build_model (*args, **kwargs)
    build model

forward (batch, mode)
    Defines the computation performed at every call.
    Should be overridden by all subclasses.
```

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

---

## Module contents

### Module contents

#### 5.1.4 crslab.model.recommendation package

##### Subpackages

##### crslab.model.recommendation.bert package

##### Submodules

##### BERT

---

## References

Devlin, Jacob, et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” in NAACL 2019.

---

**class** crslab.model.recommendation.bert.bert.**BERTModel** (*opt, device, vocab, side\_data*)  
 Bases: *crslab.model.base.BaseModel*

**item\_size**

A integer indicating the number of items.

**Parameters**

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

**build\_model()**  
 build model

**forward** (*batch, mode='train'*)  
 Defines the computation performed at every call.  
 Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### crslab.model.recommendation.gru4rec package

#### Submodules

##### GRU4REC

---

#### References

Hidasi, Balázs, et al. “Session-Based Recommendations with Recurrent Neural Networks.” in ICLR 2016.

---

```
class crslab.model.recommendation.gru4rec.gru4rec.GRU4RECMODEL(opt, device, vocab, side_data)
```

Bases: *crslab.model.base.BaseModel*

#### item\_size

A integer indicating the number of items.

#### hidden\_size

A integer indicating the hidden state size in GRU.

#### num\_layers

A integer indicating the number of GRU layers.

#### dropout\_hidden

A float indicating the dropout rate to dropout hidden state.

#### dropout\_input

A integer indicating if we dropout the input of model.

#### embedding\_dim

A integer indicating the dimension of item embedding.

#### batch\_size

A integer indicating the batch size.

#### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

#### build\_model()

build model

#### cross\_entropy(seq\_out, pos\_ids, neg\_ids, input\_mask)

#### forward(batch, mode)

#### Parameters

- **input\_ids** – padding in left, [pad, pad, id1, id2, ..., idn]
- **target\_ids** – padding in left, [pad, pad, id2, id3, ..., y]

---

```
reconstruct_input(input_ids)
    convert the padding from left to right
```

## Module contents

### crslab.model.recommendation.popularity package

#### Submodules

##### Popularity

```
class crslab.model.recommendation.popularity.PopularityModel(opt,
                                                               de-
                                                               vice,
                                                               vo-
                                                               cab,
                                                               side_data)
```

Bases: *crslab.model.base.BaseModel*

##### **item\_size**

A integer indicating the number of items.

##### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

##### **build\_model()**

build model

##### **forward**(*batch, mode*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

## Module contents

### crslab.model.recommendation.sasrec package

#### Submodules

```
class crslab.model.recommendation.sasrec.modules.Embeddings (item_size,
hidden_size,
max_seq_length,
hid-
den_dropout_prob)
```

Bases: torch.nn.modules.module.Module

Construct the embeddings from item, position, attribute.

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*input\_ids*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.recommendation.sasrec.modules.Encoder (num_hidden_layers,
hidden_size,
num_attention_heads,
hidden_dropout_prob,
hidden_act, atten-
tion_probs_dropout_prob)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*hidden\_states, attention\_mask, output\_all\_encoded\_layers=True*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.recommendation.sasrec.modules.Intermediate (hidden_size,
hidden_act, hid-
den_dropout_prob)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*input\_tensor*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.recommendation.sasrec.modules.Layer (hidden_size,  
                          num_attention_heads,  
                          hidden_dropout_prob,  
                          hidden_act,                  atten-  
                          tion_probs_dropout_prob)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*hidden\_states*, *attention\_mask*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.recommendation.sasrec.modules.LayerNorm (hidden_size, eps=1e-12)
```

Bases: torch.nn.modules.module.Module

Construct a layernorm module in the TF style (epsilon inside the square root).

**forward** (*x*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.recommendation.sasrec.modules.SASRec (hidden_dropout_prob, de-  
                          vice_initializer_range, hid-  
                          den_size, max_seq_length,  
                          item_size,  
                          num_attention_heads,  
                          atten-  
                          tion_probs_dropout_prob,  
                          hidden_act,  
                          num_hidden_layers)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**build\_model** ()

**compute\_loss** (*y\_pred*, *y*, *subset='test'*)

**cross\_entropy** (*seq\_out*, *pos\_ids*, *neg\_ids*)

**forward** (*input\_ids*, *attention\_mask*=*None*, *output\_all\_encoded\_layers*=*True*)  
Defines the computation performed at every call.  
Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
init_model()  
init_sas_weights(module)  
    Initialize the weights.  
load_model(path)  
save_model(file_name)  
  
class crslab.model.recommendation.sasrec.modules.SelfAttention(hidden_size,  
                                                               num_attention_heads,  
                                                               hid-  
                                                               den_dropout_prob,  
                                                               atten-  
                                                               tion_probs_dropout_prob)  
Bases: torch.nn.modules.module.Module  
Initializes internal Module state, shared by both nn.Module and ScriptModule.  
  
forward (input_tensor, attention_mask)  
Defines the computation performed at every call.  
Should be overridden by all subclasses.
```

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
transpose_for_scores(x)  
    Parameters x – (bs, seq_len, all_head_size)  
    Returns x.permute(0, 2, 1, 3), (bs, num_heads, seq_len, head_size)  
  
crslab.model.recommendation.sasrec.modules.gelu(x)  
Implementation of the gelu activation function.  
For information: OpenAI GPT's gelu is slightly different (and gives slightly different results):  $0.5 * x * (1 + \text{torch.tanh}(\text{math.sqrt}(2 / \text{math.pi}) * (x + 0.044715 * \text{torch.pow}(x, 3))))$  Also see https://arxiv.org/abs/1606.08415  
crslab.model.recommendation.sasrec.modules.swish(x)
```

## SASREC

---

### References

Kang, Wang-Cheng, and Julian McAuley. “Self-attentive sequential recommendation.” in ICDM 2018.

---

```
class crslab.model.recommendation.sasrec.sasrec.SASRECMODEL (opt, device, vocab,  
                                                  side_data)
```

Bases: *crslab.model.base.BaseModel*

#### **hidden\_dropout\_prob**

A float indicating the dropout rate to dropout hidden state in SASRec.

#### **initializer\_range**

A float indicating the range of parameters initiation in SASRec.

#### **hidden\_size**

A integer indicating the size of hidden state in SASRec.

#### **max\_seq\_length**

A integer indicating the max interaction history length.

#### **item\_size**

A integer indicating the number of items.

#### **num\_attention\_heads**

A integer indicating the head number in SASRec.

#### **attention\_probs\_dropout\_prob**

A float indicating the dropout rate in attention layers.

#### **hidden\_act**

A string indicating the activation function type in SASRec.

#### **num\_hidden\_layers**

A integer indicating the number of hidden layers in SASRec.

### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

```
build_model()  
build model
```

#### **forward** (*batch*, *mode*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### crslab.model.recommendation.textcnn package

#### Submodules

##### TextCNN

---

#### References

Kim, Yoon. “Convolutional Neural Networks for Sentence Classification.” in EMNLP 2014.

---

```
class crslab.model.recommendation.textcnn.textcnn.TextCNNModel(opt, device, vocab, side_data)
```

Bases: *crslab.model.base.BaseModel*

##### **movie\_num**

A integer indicating the number of items.

##### **num\_filters**

A string indicating the number of filter in CNN.

##### **embed**

A integer indicating the size of embedding layer.

##### **filter\_sizes**

A string indicating the size of filter in CNN.

##### **dropout**

A float indicating the dropout rate.

#### Parameters

- **opt** (*dict*) – A dictionary record the hyper parameters.
- **device** (*torch.device*) – A variable indicating which device to place the data and model.
- **vocab** (*dict*) – A dictionary record the vocabulary information.
- **side\_data** (*dict*) – A dictionary record the side data.

##### **build\_model()**

build model

##### **conv\_and\_pool(x, conv)**

##### **forward(batch, mode)**

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the `Module` instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

## Module contents

### Module contents

#### 5.1.5 crslab.model.utils package

##### Subpackages

##### crslab.model.utils.modules package

##### Submodules

```
class crslab.model.utils.modules.attention.SelfAttentionBatch(dim, da, alpha=0.2, dropout=0.5)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward**(*h*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

```
class crslab.model.utils.modules.attention.SelfAttentionSeq(dim, da, alpha=0.2, dropout=0.5)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward**(*h*, mask=None, return\_logits=False)

For the padding tokens, its corresponding mask is True if mask==[1, 1, 1, ...]

```
class crslab.model.utils.modules.transformer.MultiHeadAttention(n_heads, dim, dropout=0.0)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward**(query, key=None, value=None, mask=None)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

```
class crslab.model.utils.modules.transformer.TransformerDecoder(n_heads,
                                                               n_layers, embedding_size,
                                                               ffn_size, vocabulary_size,
                                                               embed-
                                                               ding=None,
                                                               dropout=0.0,
                                                               atten-
                                                               tion_dropout=0.0,
                                                               relu_dropout=0.0,
                                                               embed-
                                                               dings_scale=True,
                                                               learn_positional_embeddings=False,
                                                               padding_idx=None,
                                                               n_positions=1024)
```

Bases: torch.nn.modules.module.Module

Transformer Decoder layer.

#### Parameters

- **n\_heads** (*int*) – the number of multihead attention heads.
- **n\_layers** (*int*) – number of transformer layers.
- **embedding\_size** (*int*) – the embedding sizes. Must be a multiple of n\_heads.
- **ffn\_size** (*int*) – the size of the hidden layer in the FFN
- **embedding** – an embedding matrix for the bottom layer of the transformer. If none, one is created for this encoder.
- **dropout** (*float*) – Dropout used around embeddings and before layer layer normalizations. This is used in Vaswani 2017 and works well on large datasets.
- **attention\_dropout** (*float*) – Dropout performed after the multhead attention softmax. This is not used in Vaswani 2017.
- **padding\_idx** (*int*) – Reserved padding index in the embeddings matrix.
- **learn\_positional\_embeddings** (*bool*) – If off, sinusoidal embeddings are used. If on, position embeddings are learned from scratch.
- **embeddings\_scale** (*bool*) – Scale embeddings relative to their dimensionality. Found useful in fairseq.
- **n\_positions** (*int*) – Size of the position embeddings matrix.

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*input, encoder\_state, incr\_state=None*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

---

```
class crslab.model.utils.modules.transformer.TransformerDecoderLayer (n_heads,  

    embed-  

    ding_size,  

    ffn_size,  

    atten-  

    tion_dropout=0.0,  

    relu_dropout=0.0,  

    dropout=0.0)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*x, encoder\_output, encoder\_mask*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.utils.modules.transformer.TransformerEncoder (n_heads,  

    n_layers, em-  

    bedding_size,  

    ffn_size, vo-  

    cabulary_size,  

    embed-  

    ding=None,  

    dropout=0.0,  

    atten-  

    tion_dropout=0.0,  

    relu_dropout=0.0,  

    padding_idx=0,  

    learn_positional_embeddings=False,  

    embed-  

    dings_scale=False,  

    reduc-  

    tion=True,  

    n_positions=1024)
```

Bases: torch.nn.modules.module.Module

Transformer encoder module.

#### Parameters

- **n\_heads** (*int*) – the number of multihead attention heads.
- **n\_layers** (*int*) – number of transformer layers.
- **embedding\_size** (*int*) – the embedding sizes. Must be a multiple of n\_heads.
- **ffn\_size** (*int*) – the size of the hidden layer in the FFN
- **embedding** – an embedding matrix for the bottom layer of the transformer. If none, one is created for this encoder.
- **dropout** (*float*) – Dropout used around embeddings and before layer layer normalizations. This is used in Vaswani 2017 and works well on large datasets.

- **attention\_dropout** (*float*) – Dropout performed after the multhead attention softmax. This is not used in Vaswani 2017.
- **relu\_dropout** (*float*) – Dropout used after the ReLU in the FFN. Not used in Vaswani 2017, but used in Tensor2Tensor.
- **padding\_idx** (*int*) – Reserved padding index in the embeddings matrix.
- **learn\_positional\_embeddings** (*bool*) – If off, sinusoidal embeddings are used. If on, position embeddings are learned from scratch.
- **embeddings\_scale** (*bool*) – Scale embeddings relative to their dimensionality. Found useful in fairseq.
- **reduction** (*bool*) – If true, returns the mean vector for the entire encoding sequence.
- **n\_positions** (*int*) – Size of the position embeddings matrix.

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*input*)

input data is a FloatTensor of shape [batch, seq\_len, dim] mask is a ByteTensor of shape [batch, seq\_len], filled with 1 when inside the sequence and 0 outside.

```
class crslab.model.utils.modules.transformer.TransformerEncoderLayer(n_heads,
                                                                    embed-
                                                                    ding_size,
                                                                    ffn_size,
                                                                    atten-
                                                                    tion_dropout=0.0,
                                                                    relu_dropout=0.0,
                                                                    dropout=0.0)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*tensor, mask*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

```
class crslab.model.utils.modules.transformer.TransformerFFN(dim,      dim_hidden,
                                                               relu_dropout=0.0)
```

Bases: torch.nn.modules.module.Module

Initializes internal Module state, shared by both nn.Module and ScriptModule.

**forward** (*x*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

---

**Note:** Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

---

```
crslab.model.utils.modules.transformer._normalize(tensor, norm_layer)
    Broadcast layer norm

crslab.model.utils.modules.transformer.create_position_codes(n_pos, dim, out)

crslab.model.utils.modules.transformer.neginf(dtype)
    Returns a representable finite number near -inf for a dtype.
```

## Module contents

### Submodules

```
crslab.model.utils.functions.edge_to_pyg_format(edge, type='RGCN')
crslab.model.utils.functions.sort_for_packed_sequence(lengths: torch.Tensor)

Parameters lengths – 1D array of lengths
Returns sorted_lengths (lengths in descending order), sorted_idx (indices to sort), rev_idx (indices to retrieve original order)
```

## Module contents

## 5.2 Submodules

```
class crslab.model.base.BaseModel(opt, device, dpath=None, resource=None)
Bases: abc.ABC, torch.nn.modules.module.Module

Base class for all models

abstract build_model(*args, **kwargs)
    build model

converse(batch, mode)
    calculate loss and prediction of conversation for batch under certain mode

Parameters
    • batch(dict or tuple) – batch data
    • mode(str, optional) – train/valid/test.

guide(batch, mode)
    calculate loss and prediction of guidance for batch under certain mode

Parameters
    • batch(dict or tuple) – batch data
    • mode(str, optional) – train/valid/test.

recommend(batch, mode)
    calculate loss and prediction of recommendation for batch under certain mode

Parameters
    • batch(dict or tuple) – batch data
    • mode(str, optional) – train/valid/test.
```

## 5.3 Module contents

```
crslib.model.get_model (config, model_name, device, vocab, side_data=None)
```

---

**CHAPTER  
SIX**

---

**CRSLAB.SYSTEM PACKAGE**

**6.1 Submodules**

**6.2 Module contents**



---

CHAPTER  
**SEVEN**

---

## **INDICES AND TABLES**

- genindex
- modindex
- search



## PYTHON MODULE INDEX

### C

crslab.config, 6  
crslab.config.config, 5  
crslab.data, 22  
crslab.data.dataloader, 15  
crslab.data.dataloader.base, 7  
crslab.data.dataloader.kbrd, 9  
crslab.data.dataloader.kgsf, 10  
crslab.data.dataloader.redial, 11  
crslab.data.dataloader.tgredial, 12  
crslab.data.dataloader.utils, 14  
crslab.data.dataset, 22  
crslab.data.dataset.durecdial, 16  
crslab.data.dataset.durecdial.durecdial,  
    15  
crslab.data.dataset.durecdial.resources,  
    16  
crslab.data.dataset.gorecdial, 17  
crslab.data.dataset.gorecdial.gorecdial,  
    16  
crslab.data.dataset.gorecdial.resources,  
    17  
crslab.data.dataset.inspired, 18  
crslab.data.dataset.inspired.inspired,  
    17  
crslab.data.dataset.inspired.resources,  
    18  
crslab.data.dataset.opendialkg, 20  
crslab.data.dataset.opendialkg.opendialkg,  
    18  
crslab.data.dataset.opendialkg.resources,  
    19  
crslab.data.dataset.redial, 21  
crslab.data.dataset.redial.redial, 20  
crslab.data.dataset.redial.resources,  
    21  
crslab.data.dataset.tgredial, 22  
crslab.data.dataset.tgredial.resources,  
    21  
crslab.data.dataset.tgredial.tgredial,  
    21  
crslab.model, 58  
crslab.model.base, 57  
crslab.model.conversation, 28  
crslab.model.conversation.gpt2, 26  
crslab.model.conversation.gpt2.gpt2, 25  
crslab.model.conversation.transformer,  
    28  
crslab.model.conversation.transformer.transformer,  
    26  
crslab.model.crs, 40  
crslab.model.crs.kbrd, 30  
crslab.model.crs.kbrd.kbrd, 28  
crslab.model.crs.kgsf, 34  
crslab.model.crs.kgsf.kgsf, 30  
crslab.model.crs.kgsf.modules, 33  
crslab.model.crs.kgsf.resources, 34  
crslab.model.crs.redial, 37  
crslab.model.crs.redial.modules, 34  
crslab.model.crs.redial.redial\_conv, 35  
crslab.model.crs.redial.redial\_rec, 36  
crslab.model.crs.tgredial, 40  
crslab.model.crs.tgredial.tg\_conv, 37  
crslab.model.crs.tgredial.tg\_policy, 38  
crslab.model.crs.tgredial.tg\_rec, 39  
crslab.model.policy, 45  
crslab.model.policy.conv\_bert, 41  
crslab.model.policy.conv\_bert.conv\_bert,  
    40  
crslab.model.policy.mgcg, 42  
crslab.model.policy.mgcg.mgcg, 41  
crslab.model.policy.pmi, 43  
crslab.model.policy.pmi.pmi, 42  
crslab.model.policy.profile\_bert, 44  
crslab.model.policy.profile\_bert.profile\_bert,  
    43  
crslab.model.policy.topic\_bert, 45  
crslab.model.policy.topic\_bert.topic\_bert,  
    44  
crslab.model.recommendation, 53  
crslab.model.recommendation.bert, 46  
crslab.model.recommendation.bert.bert,  
    45  
crslab.model.recommendation.gru4rec, 47

```
crslab.model.recommendation.gru4rec.gru4rec,  
    46  
crslab.model.recommendation.popularity,  
    48  
crslab.model.recommendation.popularity.popularity,  
    47  
crslab.model.recommendation.sasrec, 52  
crslab.model.recommendation.sasrec.modules,  
    48  
crslab.model.recommendation.sasrec.sasrec,  
    50  
crslab.model.recommendation.textcnn, 53  
crslab.model.recommendation.textcnn.textcnn,  
    52  
crslab.model.utils, 57  
crslab.model.utils.functions, 57  
crslab.model.utils.modules, 57  
crslab.model.utils.modules.attention,  
    53  
crslab.model.utils.modules.transformer,  
    53
```

# INDEX

## Symbols

_normalize() (in module <code>crslab.model.utils.modules.transformer</code> ),	56	build_model() ( <code>crslab.model.conversation.gpt2.gpt2.GPT2Model</code> method), 25
_starts() ( <code>crslab.model.conversation.transformer.transformer.TransformerModel</code> method), 28		build_model() ( <code>crslab.model.conversation.transformer.transformer.TransformerModel</code> method), 28
_starts() ( <code>crslab.model.crs.kbrd.kbrd.KBRDModel</code> method), 30		build_model() ( <code>crslab.model.crs.kbrd.kbrd.KBRDModel</code> method), 30
_starts() ( <code>crslab.model.crs.kgsf.kgsf.KGSFModel</code> method), 32		build_model() ( <code>crslab.model.crs.kgsf.kgsf.KGSFModel</code> method), 32
<b>A</b>		build_model() ( <code>crslab.model.crs.redial.redial_conv.ReDialConvModel</code> method), 36
<code>add_start_end_token_idx()</code> (in module <code>crslab.data.dataloader.utils</code> ), 14		build_model() ( <code>crslab.model.crs.redial.redial_rec.ReDialRecModel</code> method), 37
attention_dropout ( <code>crslab.model.conversation.transformer.transformer.TransformerModel</code> attribute), 27		build_model() ( <code>crslab.model.crs.tgredial.tg_conv.TGConvModel</code> method), 38
attention_dropout ( <code>crslab.model.crs.kbrd.kbrd.KBRDModel</code> attribute), 29		build_model() ( <code>crslab.model.crs.tgredial.tg_policy.TGPolicyModel</code> method), 39
attention_dropout ( <code>crslab.model.crs.kgsf.kgsf.KGSFModel</code> attribute), 31		build_model() ( <code>crslab.model.crs.tgredial.tg_rec.TGRecModel</code> method), 40
attention_probs_dropout_prob ( <code>crslab.model.crs.tgredial.tg_rec.TGRecModel</code> attribute), 40		build_model() ( <code>crslab.model.policy.conv_bert.conv_bert.ConvBERTModel</code> method), 41
attention_probs_dropout_prob ( <code>crslab.model.recommendation.sasrec.sasrec.SASRECModel</code> attribute), 51		build_model() ( <code>crslab.model.policy.mgcg.mgcg.MGCCModel</code> method), 42
<b>B</b>		build_model() ( <code>crslab.model.policy.pmi.pmi.PMIModel</code> method), 43
<code>BaseDataLoader</code> (class in <code>crslab.data.dataloader.base</code> ), 7	in	build_model() ( <code>crslab.model.policy.profile_bert.profile_bert.ProfileBERTModel</code> method), 43
<code> BaseModel</code> (class in <code>crslab.model.base</code> ), 57		build_model() ( <code>crslab.model.policy.topic_bert.topic_bert.TopicBERTModel</code> method), 44
<code>batch_size</code> ( <code>crslab.model.recommendation.gru4rec.gru4rec.GRU4RECModel</code> attribute), 46		build_model() ( <code>crslab.model.recommendation.popularity.popularity.PopularityModel</code> method), 47
<code>BERTModel</code> (class in <code>crslab.model.recommendation.bert.bert</code> ), 45	in	build_model() ( <code>crslab.model.recommendation.sasrec.modules.SASRecModel</code> method), 49
<code>build_model()</code> ( <code>crslab.model.base.BaseModel</code> method), 57		build_model() ( <code>crslab.model.recommendation.sasrec.sasrec.SASRECModel</code> method), 51
		build_model() ( <code>crslab.model.recommendation.textcnn.textcnn.TextCNNModel</code> method), 52

C

- `cross_entropy()` (*crslab.model.recommendation.sasrec.modules.SASRecMethod*), 49
- `calculate_loss()` (*crslab.model.conversation.gpt2.gpt2.GPT2ModelMethod*), 25
- `calculate_loss()` (*crslab.model.crs.tgredial.tg\_conv.TGConvModelMethod*), 38
- `compute_loss()` (*crslab.model.recommendation.sasrec.modules.SASRecModule*), 49
- `Config` (class in *crslab.config.config*), 5
- `context_truncate` (*crslab.model.conversation.gpt2.gpt2.GPT2ModelModule*), 15
- `context_truncate` (*crslab.model.crs.tgredial.tg\_conv.TGConvModelModule*), 7
- `conv_and_pool()` (*crslab.model.recommendation.textcnn.textcnn.TextCNNModelModule*), 9
- `conv_batchify()` (*crslab.data.dataloader.base.BaseDataLoaderModule*), 7
- `conv_batchify()` (*crslab.data.dataloader.kbrd.KBRDDataloaderModule*), 11
- `conv_batchify()` (*crslab.data.dataloader.kgsf.KGSFDataLoaderModule*), 10
- `conv_batchify()` (*crslab.data.dataloader.redial.ReDialDataLoaderModule*), 11
- `conv_batchify()` (*crslab.data.dataloader.tgredial.TGRedialDataLoaderModule*), 13
- `conv_interact()` (*crslab.data.dataloader.base.BaseDatasetModule*), 16
- `conv_interact()` (*crslab.data.dataloader.tgredial.TGRedialDatasetModule*), 15
- `conv_process_fn()` (*crslab.data.dataloader.base.BaseDatasetModule*), 7
- `conv_process_fn()` (*crslab.data.dataloader.kbrd.KBRDDatasetLoaderModule*), 9
- `conv_process_fn()` (*crslab.data.dataloader.kgsf.KGSFDataLoaderModule*), 10
- `conv_process_fn()` (*crslab.data.dataloader.redial.ReDialDatasetModule*), 12
- `ConvBERTModel` (class in *crslab.model.policy.conv\_bert.conv\_bert*), 40
- `converse()` (*crslab.model.base.BaseModelModule*), 57
- `converse()` (*crslab.model.crs.kbrd.kbrd.KBRDModelModule*), 30
- `converse()` (*crslab.model.crs.kgsf.kgsf.KGSFModelModule*), 32
- `create_position_codes()` (in *module crslab.model.utils.modules.transformer*), 57
- `cross_entropy()` (*crslab.model.recommendation.gru4rec.gru4rec.GRU4RECMModelModule*), 21

```

crslab.data.dataset.tgredial
    module, 22
crslab.data.dataset.tgredial.resources
    module, 21
crslab.data.dataset.tgredial.tgredial
    module, 21
crslab.model
    module, 58
crslab.model.base
    module, 57
crslab.model.conversation
    module, 28
crslab.model.conversation.gpt2
    module, 26
crslab.model.conversation.gpt2.gpt2
    module, 25
crslab.model.conversation.transformer
    module, 28
crslab.model.conversation.transformer.transfformmodel.recommendation
    module, 26
crslab.model.crs
    module, 40
crslab.model.crs.kbrd
    module, 30
crslab.model.crs.kbrd.kbrd
    module, 28
crslab.model.crs.kgsf
    module, 34
crslab.model.crs.kgsf.kgsf
    module, 30
crslab.model.crs.kgsf.modules
    module, 33
crslab.model.crs.kgsf.resources
    module, 34
crslab.model.crs.redial
    module, 37
crslab.model.crs.redial.modules
    module, 34
crslab.model.crs.redial.redial_conv
    module, 35
crslab.model.crs.redial.redial_rec
    module, 36
crslab.model.crs.tgredial
    module, 40
crslab.model.crs.tgredial.tg_conv
    module, 37
crslab.model.crs.tgredial.tg_policy
    module, 38
crslab.model.crs.tgredial.tg_rec
    module, 39
crslab.model.policy
    module, 45
crslab.model.policy.conv_bert
    module, 41
crslab.model.policy.conv_bert.conv_bert
    module, 40
crslab.model.policy.mgcg
    module, 42
crslab.model.policy.mgcg.mgcg
    module, 41
crslab.model.policy.pmi
    module, 43
crslab.model.policy.pmi.pmi
    module, 42
crslab.model.policy.profile_bert
    module, 44
crslab.model.policy.profile_bert.profile_bert
    module, 43
crslab.model.policy.topic_bert
    module, 45
crslab.model.policy.topic_bert.topic_bert
    module, 44
crslab.model.recommendation
    module, 53
crslab.model.recommendation.bert
    module, 46
crslab.model.recommendation.bert.bert
    module, 45
crslab.model.recommendation.gru4rec
    module, 47
crslab.model.recommendation.gru4rec.gru4rec
    module, 46
crslab.model.recommendation.popularity
    module, 48
crslab.model.recommendation.popularity.popularity
    module, 47
crslab.model.recommendation.sasrec
    module, 52
crslab.model.recommendation.sasrec.modules
    module, 48
crslab.model.recommendation.sasrec.sasrec
    module, 50
crslab.model.recommendation.textcnn
    module, 53
crslab.model.recommendation.textcnn.textcnn
    module, 52
crslab.model.utils
    module, 57
crslab.model.utils.functions
    module, 57
crslab.model.utils.modules
    module, 57
crslab.model.utils.modules.attention
    module, 53
crslab.model.utils.modules.transformer
    module, 53

```

**D**

dataloader\_register\_table (in module `crslab.data`), 22  
 dataset\_language\_map (in module `crslab.data`), 22  
 DATASET\_PATH (in module `crslab.config`), 6  
 dataset\_register\_table (in module `crslab.data`), 22  
 decode\_beam\_search () (crslab.model.crs.kbrd.kbrd.KBRDModel method), 30  
 decode\_forced () (crslab.model.crs.kbrd.kbrd.KBRDModel method), 30  
 decode\_greedy () (crslab.model.crs.kbrd.kbrd.KBRDModel method), 30  
 decoder\_embedding\_dim (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 36  
 decoder\_hidden\_size (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 36  
 decoder\_num\_layers (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 36  
 dialog\_encoder\_hidden\_size (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 36  
 dialog\_encoder\_num\_layers (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 36  
 dropout (crslab.model.conversation.transformer.transformer.TransformerModel attribute), 27  
 dropout (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 29  
 dropout (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 31  
 dropout (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 36  
 dropout (crslab.model.recommendation.textcnn.TextCNNModel attribute), 52  
 dropout\_hidden (crslab.model.policy.mgcg.MGCGModel method), 26  
 dropout\_hidden (crslab.model.recommendation.gru4rec.GRU4RECModel attribute), 46  
 dropout\_input (crslab.model.recommendation.gru4rec.GRU4RECModel attribute), 46  
 DuRecDialDataset (class in `crslab.data.dataset.durec dial.durec dial`), 15

**E**

edge\_to\_pyg\_format () (in module `crslab.model.utils.functions`), 57  
 embed (crslab.model.recommendation.textcnn.TextCNNModel attribute), 52

embedding\_dim (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 35  
 embedding\_dim (crslab.model.policy.mgcg.mgcg.MGCGModel attribute), 41  
 embedding\_dim (crslab.model.recommendation.gru4rec.gru4rec.GRU4RECModel attribute), 46  
 EMBEDDING\_PATH (in module `crslab.config`), 6  
 Embeddings (class in `crslab.model.recommendation.sasrec.modules`), 48  
 embeddings\_scale (crslab.model.conversation.transformer.transformer.TransformerModel attribute), 27  
 embeddings\_scale (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 29  
 embeddings\_scale (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 32  
 encode\_user () (crslab.model.crs.kbrd.kbrd.KBRDModel method), 30  
 Encoder (class in `crslab.model.recommendation.sasrec.modules`), 48  
 end\_token\_idx (crslab.model.conversation.transformer.transformer.TransformerModel attribute), 26  
 end\_token\_idx (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 28  
 end\_token\_idx (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 31  
 end\_token\_idx (crslab.model.crs.redial.redial\_conv.ReDialConvModel attribute), 35

**F**  
 ffn\_size (crslab.model.conversation.transformer.transformer.TransformerModel attribute), 27  
 ffn\_size (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 29  
 ffn\_size (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 31

filter\_sizes (crslab.model.recommendation.textcnn.TextCNNModel attribute), 52  
 forward () (crslab.model.conversation.gpt2.GPT2Model method), 52  
 forward () (crslab.model.conversation.transformer.transformer.TransformerModel method), 27  
 forward () (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 29  
 forward () (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 31  
 forward () (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 32  
 forward () (crslab.model.crs.kgsf.modules.GateLayer method), 33  
 forward () (crslab.model.crs.kgsf.modules.TransformerDecoderKG method), 34

forward () (crslab.model.crs.kgsf.modules.TransformerDecoderLayerKG method), 34  
 forward () (crslab.model.crs.kgsf.modules.TransformerDecoderLayerKG method), 34  
 forward () (crslab.model.crs.redial.modules.HRNN method), 34

forward() (crslab.model.crs.redial.modules.SwitchingDecoderForward() (crslab.model.utils.modules.transformer.TransformerDecoder  
     method), 35  
 forward() (crslab.model.crs.redial.redial\_conv.ReDialConvForward() (crslab.model.utils.modules.transformer.TransformerEncoder  
     method), 36  
 forward() (crslab.model.crs.redial.redial\_rec.ReDialRecForward() (crslab.model.utils.modules.transformer.TransformerEncoder  
     method), 37  
 forward() (crslab.model.crs.tgredial.tg\_conv.TGConvForward() (crslab.model.utils.modules.transformer.TransformerFFN  
     method), 38  
 forward() (crslab.model.crs.tgredial.tg\_policy.TGPolicyForward() (crslab.model.crs.kbrd.kbrd.KBRDModel  
     method), 39  
 forward() (crslab.model.crs.tgredial.tg\_rec.TGRecForward() (crslab.model.crs.kgsf.kgsf.KGSFModel  
     method), 40  
 forward() (crslab.model.policy.conv\_bert.conv\_bert.ConvBERTForward() (crslab.model.crs.kgsf.kgsf.KGSFModel  
     method), 41  
 forward() (crslab.model.policy.mgcg.mgcg.MGCGModel  
     method), 42  
 forward() (crslab.model.policy.pmi.pmi.PMIModel  
     method), 43  
 forward() (crslab.model.policy.profile\_bert.profile\_bert.ProfileBERTModel  
     method), 44  
 forward() (crslab.model.policy.topic\_bert.topic\_bert.TopicBERTModel  
     method), 44  
 forward() (crslab.model.recommendation.bert.bert.BERTModel  
     method), 45  
 forward() (crslab.model.recommendation.gru4rec.gru4rec.GRU4RECMODEL  
     method), 46  
 forward() (crslab.model.recommendation.popularity.PopularityModel  
     method), 47  
 forward() (crslab.model.recommendation.sasrec.modules.Embeddings  
     method), 48  
 forward() (crslab.model.recommendation.sasrec.modules.Encoder  
     method), 48  
 forward() (crslab.model.recommendation.sasrec.modules.IntermediateEncoder  
     method), 48  
 forward() (crslab.model.recommendation.sasrec.modules.LayerNorm  
     method), 49  
 forward() (crslab.model.recommendation.sasrec.modules.SASRec  
     method), 49  
 forward() (crslab.model.recommendation.sasrec.modules.SelfAttention  
     method), 50  
 forward() (crslab.model.recommendation.sasrec.sasrec.SASRECMODEL  
     method), 51  
 forward() (crslab.model.recommendation.textcnn.textcnn.TextCNNModel  
     method), 52  
 forward() (crslab.model.utils.modules.attention.SelfAttentionBatch  
     method), 53  
 forward() (crslab.model.utils.modules.attention.SelfAttentionSeq  
     method), 53  
 forward() (crslab.model.utils.modules.transformer.MultiHeadAttention  
     method), 53  
 forward() (crslab.model.utils.modules.transformer.TransformerDecoder  
     method), 54

**G**

GateLayer (class in crslab.model.crs.kgsf.modules), 33  
 gelu() (in module crslab.model.recommendation.sasrec.modules),  
 generate() (crslab.model.conversation.gpt2.gpt2.GPT2Model  
     method), 26  
 generate() (crslab.model.crs.tgredial.tg\_conv.TGConvModel  
     method), 38  
 generate\_bs() (crslab.model.conversation.gpt2.gpt2.GPT2Model  
     method), 26  
 generate\_bs() (crslab.model.crs.tgredial.tg\_conv.TGConvModel  
     method), 38  
 get() (crslab.config.config.Config method), 5  
 get\_CoVData() (crslab.data.dataloader.base.BaseDataLoader  
     method), 7  
 get\_Data() (crslab.data.dataloader.base.BaseDataLoader  
     method), 8  
 get\_dataset() (in module crslab.data), 22  
 get\_onehot() (in module crslab.data), 22  
 get\_layerlength() (crslab.model.policy.mgcg.mgcg.MGCGModel  
     method), 42  
 get\_layerNorm() (in module crslab.model), 58  
 get\_policy\_data() (crslab.data.dataloader.base.BaseDataLoader  
     method), 8  
 get\_pretain\_data() (crslab.data.dataloader.kgsf.KGSFDataLoader  
     method), 11  
 get\_rec\_data() (crslab.data.dataloader.base.BaseDataLoader  
     method), 8  
 get\_utterance\_encoding() (crslab.model.crs.redial.modules.HRNN  
     method), 35  
 GORECDIALDataset (class in crslab.data.dataset.gorecdial.gorecdial),  
 in

GPT2Model (class in `crslab.model.conversation.gpt2.gpt2`, 25)

GRU4RECMODEL (class in `crslab.model.recommendation.gru4rec.gru4rec`, 46)

guide () (`crslab.model.base.BaseModel` method), 57

**H**

hidden\_act (`crslab.model.crs.tgredial.tg_rec.TGRecModel` attribute), 27  
attribute), 40

hidden\_act (`crslab.model.recommendation.sasrec.sasrec.SASRECModel` attribute), 51

hidden\_dropout\_prob (`crslab.model.crs.tgredial.tg_rec.TGRecModel` attribute), 39

hidden\_dropout\_prob (`crslab.model.recommendation.sasrec.sasrec.SASRECModel` attribute), 51

hidden\_size (`crslab.model.crs.tgredial.tg_rec.TGRecModel` attribute), 39

hidden\_size (`crslab.model.policy.mgcg.mgcg.MGCGModel` attribute), 41

hidden\_size (`crslab.model.recommendation.gru4rec.gru4rec.GRU4RECMODEL` attribute), 46

hidden\_size (`crslab.model.recommendation.sasrec.sasrec.SASRECModel` attribute), 51

HRNN (class in `crslab.model.crs.redial.modules`), 34

**I**

init\_model () (`crslab.model.recommendation.sasrec.modules.SASRec` method), 50

init\_sas\_weights () (`crslab.model.recommendation.sasrec.modules.SASRec` method), 50

initializer\_range (`crslab.model.crs.tgredial.tg_rec.TGRecModel` attribute), 39

initializer\_range (`crslab.model.recommendation.sasrec.sasrec.SASRECModel` attribute), 51

InspiredDataset (class in `crslab.data.dataset.inspired.inspired`), 17

Intermediate (class in `crslab.model.recommendation.sasrec.modules`), 48

item\_size (`crslab.model.crs.tgredial.tg_rec.TGRecModel` attribute), 39

item\_size (`crslab.model.recommendation.bert.bert.BERTModel` attribute), 45

item\_size (`crslab.model.recommendation.gru4rec.gru4rec.GRU4RECMODEL` attribute), 46

item\_size (`crslab.model.recommendation.popularity.popularity.PopularityModel` attribute), 47

**K**

KBRDDataloader (class in `crslab.data.dataloader.kbrd`), 9

KBRDModel (class in `crslab.model.crs.kbrd.kbrd`), 28

kg\_emb\_dim (`crslab.model.conversation.transformer.transformer.Transfor` attribute), 27  
attribute), 40

kg\_emb\_dim (`crslab.model.crs.kbrd.kbrd.KBRDModel` attribute), 31

KGSFDataLoader (class in `crslab.data.dataloader.kgsf`), 10

KGSFModel (class in `crslab.model.crs.kgsf.kgsf`), 31

**L**

layer (class in `crslab.model.recommendation.sasrec.modules`), 49

layer\_sizes (`crslab.model.crs.redial.redial_rec.ReDialRecModel` attribute), 37

LayerGRU4RECMODEL (class in `crslab.model.recommendation.sasrec.modules`), 46

learn\_positional\_embeddings (`crslab.model.conversation.transformer.transformer.Transfor` attribute), 27

learn\_positional\_embeddings

**M**

load\_model () (`crslab.model.recommendation.sasrec.modules.SASRec` method), 50

load\_yaml\_configs () (`crslab.config.config.Config` static method), 5

longest\_label (`crslab.model.crs.kbrd.kbrd.KBRDModel` attribute), 29

max\_seq\_length (`crslab.model.crs.tgredial.tg_rec.TGRecModel` attribute), 39

max\_seq\_length (`crslab.model.recommendation.sasrec.sasrec.SASRECModel` attribute), 51

merge\_utt () (in module `crslab.data.dataloader.utils`), 14

merge\_utt\_replace () (in module `crslab.data.dataloader.utils`), 14

MGCGModel (class in `crslab.model.policy.mgcg.mgcg`), 41

MODEL\_PATH (*in module crslab.config*), 6  
 module  
   crslab.config, 6  
   crslab.config.config, 5  
   crslab.data, 22  
   crslab.data.dataloader, 15  
   crslab.data.dataloader.base, 7  
   crslab.data.dataloader.kbrd, 9  
   crslab.data.dataloader.kgsf, 10  
   crslab.data.dataloader.redial, 11  
   crslab.data.dataloader.tgredial, 12  
   crslab.data.dataloader.utils, 14  
   crslab.data.dataset, 22  
   crslab.data.dataset.durecdial, 16  
   crslab.data.dataset.durecdial.durecdial, 15  
   crslab.data.dataset.durecdial.resources, 16  
   crslab.data.dataset.gorecdial, 17  
   crslab.data.dataset.gorecdial.gorecdial, 16  
   crslab.data.dataset.gorecdial.resources, 17  
   crslab.data.dataset.inspired, 18  
   crslab.data.dataset.inspired.inspired, 17  
   crslab.data.dataset.inspired.resources, 18  
   crslab.data.dataset.opendialkg, 20  
   crslab.data.dataset.opendialkg.opendialkg, 18  
   crslab.data.dataset.opendialkg.resources, 19  
   crslab.data.dataset.redial, 21  
   crslab.data.dataset.redial.redial, 20  
   crslab.data.dataset.redial.resources, 21  
   crslab.data.dataset.tgredial, 22  
   crslab.data.dataset.tgredial.resources, 21  
   crslab.data.dataset.tgredial.tgredial, 21  
   crslab.model, 58  
   crslab.model.base, 57  
   crslab.model.conversation, 28  
   crslab.model.conversation.gpt2, 26  
   crslab.model.conversation.gpt2.gpt2, 25  
   crslab.model.conversation.transformer, 28  
   crslab.model.conversation.transformer.transførmer, 26  
   crslab.model.crs, 40  
 crslab.model.crs.kbrd, 30  
 crslab.model.crs.kbrd.kbrd, 28  
 crslab.model.crs.kgsf, 34  
 crslab.model.crs.kgsf.kgsf, 30  
 crslab.model.crs.kgsf.modules, 33  
 crslab.model.crs.kgsf.resources, 34  
 crslab.model.crs.redial, 37  
 crslab.model.crs.redial.modules, 34  
 crslab.model.crs.redial.redial\_conv, 35  
 crslab.model.crs.redial.redial\_rec, 36  
 crslab.model.crs.tgredial, 40  
 crslab.model.crs.tgredial.tg\_conv, 37  
 crslab.model.crs.tgredial.tg\_policy, 38  
 crslab.model.crs.tgredial.tg\_rec, 39  
 crslab.model.policy, 45  
 crslab.model.policy.conv\_bert, 41  
 crslab.model.policy.conv\_bert.conv\_bert, 40  
 crslab.model.policy.mgcg, 42  
 crslab.model.policy.mgcg.mgcg, 41  
 crslab.model.policy.pmi, 43  
 crslab.model.policy.pmi.pmi, 42  
 crslab.model.policy.profile\_bert, 44  
 crslab.model.policy.profile\_bert.profile\_bert, 43  
 crslab.model.policy.topic\_bert, 45  
 crslab.model.policy.topic\_bert.topic\_bert, 44  
 crslab.model.recommendation, 53  
 crslab.model.recommendation.bert, 46  
 crslab.model.recommendation.bert.bert, 45  
 crslab.model.recommendation.gru4rec, 47  
 crslab.model.recommendation.gru4rec.gru4rec, 46  
 crslab.model.recommendation.popularity, 48  
 crslab.model.recommendation.popularity.populari, 47  
 crslab.model.recommendation.sasrec, 52  
 crslab.model.recommendation.sasrec.modules, 48  
 crslab.model.recommendation.sasrec.sasrec, 50  
 crslab.model.recommendation.textcnn, 52  
 crslab.model.recommendation.textcnn.textcnn, 52

```

crslab.model.utils, 57
crslab.model.utils.functions, 57
crslab.model.utils.modules, 57
crslab.model.utils.modules.attention,
    53
crslab.model.utils.modules.transformer, 53
num_bases (crslab.model.conversation.transformer.transformer.Transform-
    er).num_bases (crslab.model.conversation.transformer.transformer.Transform-
    er), 27
movie_num (crslab.model.recommendation.textcnn.textcnn.TextCNNModel),
    attribute), 52
MultiHeadAttention (class crslab.model.utils.modules.transformer),
    53
in num_bases (crslab.model.crs.kgsf.kgsf.KGSFModel at-
    tribute), 31
num_filters (crslab.model.recommendation.textcnn.textcnn.TextCNNModel),
    attribute), 52
num_hidden_layers
n_entity (crslab.model.conversation.transformer.TransformerModel).crs.tgredial.tg_rec.TGRecModel
    attribute), 27
n_entity (crslab.model.crs.kbrd.kbrd.KBRDModel at-
    tribute), 29
n_entity (crslab.model.crs.kgsf.kgsf.KGSFModel at-
    tribute), 31
n_entity (crslab.model.crs.redial.redial_rec.ReDialRecModel
    attribute), 37
n_heads (crslab.model.conversation.transformer.TransformerModel),
    attribute), 27
n_heads (crslab.model.crs.kbrd.kbrd.KBRDModel at-
    tribute), 29
n_heads (crslab.model.crs.kgsf.kgsf.KGSFModel at-
    tribute), 31
n_layers (crslab.model.conversation.transformer.transformer.Transform-
    er).TransformerModel
    attribute), 27
n_layers (crslab.model.crs.kbrd.kbrd.KBRDModel at-
    tribute), 29
n_layers (crslab.model.crs.kgsf.kgsf.KGSFModel at-
    tribute), 31
n_positions (crslab.model.conversation.transformer.TransformerModel
    attribute), 27
n_positions (crslab.model.crs.kbrd.kbrd.KBRDModel
    attribute), 29
n_positions (crslab.model.crs.kgsf.kgsf.KGSFModel
    attribute), 32
n_relation (crslab.model.crs.kbrd.kbrd.KBRDModel
    attribute), 29
n_sent (crslab.model.policy.mgcg.mgcg.MGCGModel
    attribute), 42
n_sent (crslab.model.policy.profile_bert.profile_bert.ProfileBERTModel
    attribute), 43
n_word (crslab.model.conversation.transformer.TransformerModel
    attribute), 27
n_word (crslab.model.crs.kgsf.kgsf.KGSFModel at-
    tribute), 31
neginf() (in module crslab.model.utils.modules.transformer),
    57
num_attention_heads
(crslab.model.crs.tgredial.tg_rec.TGRecModel
attribute), 39
num_attention_heads
(crslab.model.recommendation.sasrec.sasrec.SASRECModel
attribute), 51
num_bases (crslab.model.conversation.transformer.transformer.Transform-
er).num_bases (crslab.model.conversation.transformer.transformer.Transform-
er), 27
TextCNNModel (crslab.model.crs.kbrd.kbrd.KBRDModel
attribute), 29
in num_bases (crslab.model.crs.kgsf.kgsf.KGSFModel at-
tribute), 31
num_filters (crslab.model.recommendation.textcnn.textcnn.TextCNNModel),
    attribute), 52
num_hidden_layers
n_entity (crslab.model.conversation.transformer.TransformerModel).crs.tgredial.tg_rec.TGRecModel
    attribute), 40
n_entity (crslab.model.crs.kbrd.kbrd.KBRDModel at-
    tribute), 42
n_entity (crslab.model.crs.redial.redial_rec.ReDialRecModel
    attribute), 42
n_heads (crslab.model.recommendation.gru4rec.gru4rec.GRU4RECModel
    attribute), 16
n_layers (crslab.model.conversation.transformer.TransformerModel)
O
OpenDialKGDataset (class crslab.data.dataset.opendialkg.opendialkg),
    18
pad_entity_idx (crslab.model.conversation.transformer.transformer.Transform-
    er).TransformerModel
    attribute), 27
pad_entity_idx (crslab.model.crs.kgsf.kgsf.KGSFModel
    attribute), 31
pad_entity_idx (crslab.model.crs.redial.redial_rec.ReDialRecModel
    attribute), 35
pad_id (crslab.model.conversation.gpt2.gpt2.GPT2Model
    attribute), 25
pad_id (crslab.model.crs.tgredial.tg_conv.TGConvModel
    attribute), 38
pad_token_idx (crslab.model.conversation.transformer.transformer.Transform-
    er).TransformerModel
    attribute), 26
pad_token_idx (crslab.model.crs.kbrd.kbrd.KBRDModel
    attribute), 28
pad_token_idx (crslab.model.crs.kgsf.kgsf.KGSFModel
    attribute), 31
pad_token_idx (crslab.model.crs.redial.redial_conv.ReDialConvModel
    attribute), 35
pad_topic (crslab.model.policy.pmi.pmi.PMIModel
    attribute), 42
pad_word_idx (crslab.model.conversation.transformer.transformer.Transform-
    er).TransformerModel
    attribute), 27
pad_word_idx (crslab.model.crs.kgsf.kgsf.KGSFModel
    attribute), 31

```

```

padded_tensor()           (in module crslab.data.dataloader.utils), 14
PMIModel (class in crslab.model.policy.pmi.pmi), 42
policy_batchify()
    (crslab.data.dataloader.base.BaseDataLoader method), 8
policy_batchify()
    (crslab.data.dataloader.kbrd.KBRDDataloader method), 9
policy_batchify()
    (crslab.data.dataloader.kgsf.KGSFDataLoader method), 11
policy_batchify()
    (crslab.data.dataloader.redial.ReDialDataLoader method), 12
policy_batchify()
    (crslab.data.dataloader.tgredial.TGReDialDataLoader method), 13
policy_process_fn()
    (crslab.data.dataloader.base.BaseDataLoader method), 8
policy_process_fn()
    (crslab.data.dataloader.tgredial.TGReDialDataLoader method), 13
PopularityModel (class in crslab.model.recommendation.popularity.popularity), 47
pretrain_batchify()
    (crslab.data.dataloader.kgsf.KGSFDataLoader method), 11
pretrain_embedding
    (crslab.model.conversation.transformer.transformer attribute), 27
pretrain_embedding
    (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 29
pretrain_embedding
    (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 31
pretrain_infomax()
    (crslab.model.crs.kgsf.kgsf.KGSFModel method), 32
PRETRAIN_PATH (in module crslab.config), 6
pretrained_embedding
    (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 32
pretrained_embedding
    (crslab.model.crs.redial.redial_conv.ReDialConvModel attribute), 35
ProfileBERTModel (class in crslab.model.profile_bert.profile_bert), 43

```

**R**

```

rec_batchify() (crslab.data.dataloader.base.BaseDataLoader method), 8
rec_batchify() (crslab.data.dataloader.kbrd.KBRDDataloader method), 10
rec_batchify() (crslab.data.dataloader.kgsf.KGSFDataLoader method), 11
rec_batchify() (crslab.data.dataloader.redial.ReDialDataLoader method), 12
rec_batchify() (crslab.data.dataloader.tgredial.TGReDialDataLoader method), 13
rec_interact()
    (crslab.data.dataloader.base.BaseDataLoader method), 8
rec_interact()
    (crslab.data.dataloader.tgredial.TGReDialDataLoader method), 13
rec_process_fn()
    (crslab.data.dataloader.base.BaseDataLoader method), 9
rec_process_fn()
    (crslab.data.dataloader.kbrd.KBRDDataloader method), 10
rec_process_fn()
    (crslab.data.dataloader.kgsf.KGSFDataLoader method), 11
rec_process_fn()
    (crslab.data.dataloader.redial.ReDialDataLoader method), 12
rec_process_fn()
    (crslab.data.dataloader.tgredial.TGReDialDataLoader method), 13
recommend() (crslab.model.base.BaseModel method), 57
recommend() (crslab.model.crs.kbrd.kbrd.KBRDModel method), 30
recommend() (crslab.model.crs.kgsf.kgsf.KGSFModel method), 32
reconstruct_input()
    (crslab.model.recommendation.gru4rec.gru4rec.GRU4RECMODEL method), 46
ReDialConvModel (class in crslab.model.crs.redial.redial_conv), 35
ReDialDataLoader (class in crslab.data.dataloader.redial), 11
ReDialDataset (class in crslab.data.dataset.redial.redial), 20
ReDialRecModel (class in crslab.model.crs.redial.redial_rec), 37
reduction (crslab.model.conversation.transformer.transformer.Transform attribute), 27
reduction (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 29
reduction (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 32
relu_dropout (crslab.model.conversation.transformer.transformer.Trans attribute), 27
relu_dropout (crslab.model.crs.kbrd.kbrd.KBRDModel attribute), 29
relu_dropout (crslab.model.crs.kgsf.kgsf.KGSFModel attribute), 31

```

response\_truncate  
 (crslab.model.conversation.gpt2.gpt2.GPT2Model  
 attribute), 25

response\_truncate  
 (crslab.model.crs.tgredial.tg\_conv.TGConvModel  
 attribute), 38

retain\_recommender\_target ()  
 (crslab.data.dataloader.base.BaseDataLoader  
 method), 9

**S**

SASRec (class in crslab.model.recommendation.sasrec.modules), 49

SASREModel (class in crslab.model.recommendation.sasrec.sasrec), 51

save\_model () (crslab.model.recommendation.sasrec.modules.SASREModel  
 method), 50

SAVE\_PATH (in module crslab.config), 6

SelfAttention (class in crslab.model.recommendation.sasrec.modules), 50

SelfAttentionBatch (class in crslab.model.utils.modules.attention), 53

SelfAttentionSeq (class in crslab.model.utils.modules.attention), 53

sort\_for\_packed\_sequence () (in module crslab.model.utils.functions), 57

start\_token\_idx (crslab.model.conversation.transfomer.transfomerModel  
 attribute), 26

start\_token\_idx (crslab.model.crs.kbrd.KBRDModel  
 attribute), 28

start\_token\_idx (crslab.model.crs.kgsf.KGSFModel  
 attribute), 31

start\_token\_idx (crslab.model.crs.redial\_conv.ReDialConvModel  
 attribute), 35

swish () (in module crslab.model.recommendation.sasrec.modules), 50

SwitchingDecoder (class in crslab.model.crs.redial.modules), 35

**T**

test\_data (crslab.data.dataset.durecdial.DuRecDialDataset  
 attribute), 15

test\_data (crslab.data.dataset.gorecdial.GoRecDialDataset  
 attribute), 16

test\_data (crslab.data.dataset.inspired.InspiredDataset  
 attribute), 18

test\_data (crslab.data.dataset.opendialkg.OpenDialKGDataset  
 attribute), 19

test\_data (crslab.data.dataset.redial.redial.ReDialDataset  
 attribute), 20

test\_data (crslab.data.dataset.tgredial.TGReDialDataset  
 attribute), 21

TextCNNModel (class in crslab.model.recommendation.textcnn.textcnn), 52

TGConvModel (class in crslab.model.crs.tgredial.tg\_conv), 38

TGPolicyModel (class in crslab.model.crs.tgredial.tg\_policy), 39

TGRecModel (class in crslab.model.crs.tgredial.tg\_rec), 39

TGReDialDataset (class in crslab.data.dataloader.tgredial), 12

token\_emb\_dim (crslab.model.conversation.transfomer.transfomer.TransfomerModel  
 attribute), 27

token\_emb\_dim (crslab.model.crs.kbrd.KBRDModel  
 attribute), 29

token\_emb\_dim (crslab.model.crs.kgsf.KGSFModel  
 attribute), 31

topic\_class\_num (crslab.model.policy.conv\_bert.conv\_bert.ConvBERT  
 attribute), 41

topic\_class\_num (crslab.model.policy.mgcg.MGCModel  
 attribute), 41

topic\_class\_num (crslab.model.policy.pmi.PMIModel  
 attribute), 42

topic\_class\_num (crslab.model.policy.profile\_bert.profile\_bert.ProfileBERT  
 attribute), 43

TopicBERTModel (class in crslab.model.policy.topic\_bert.topic\_bert), 44

TopicBERTModel (class in crslab.model.topic\_bert.topic\_bert), 44

train\_data (crslab.data.dataset.gorecdial.gorecdial.GoRecDialDataset  
 attribute), 16

train\_data (crslab.data.dataset.inspired.InspiredDataset  
 attribute), 17

train\_data (crslab.data.dataset.opendialkg.opendialkg.OpenDialKGDataset  
 attribute), 19

train\_data (crslab.data.dataset.redial.redial.ReDialDataset  
 attribute), 20

train\_data (crslab.data.dataset.tgredial.TGReDialDataset  
 attribute), 21

TransformerDecoder (class in crslab.model.utils.modules.transformer), 53

TransformerDecoderKG (class in crslab.model.crs.kgsf.modules), 33

TransformerDecoderLayer (class in crslab.model.utils.modules.transformer), 54

```

TransformerDecoderLayerKG      (class      in  vocab (crslab.data.dataset.redial.ReDialDataset
      crslab.model.crs.kgsf.modules), 34      attribute), 20
TransformerEncoder            (class      in  vocab (crslab.data.dataset.tgredial.TGReDialDataset
      crslab.model.utils.modules.transformer), 55      attribute), 21
TransformerEncoderLayer       (class      in  vocab_size (crslab.model.conversation.transformer.Transfor
      crslab.model.utils.modules.transformer), 56      mation attribute), 26
TransformerFFN               (class      in  vocab_size (crslab.model.crs.kbrd.kbrd.KBRDModel
      crslab.model.utils.modules.transformer), 56      attribute), 28
TransformerModel              (class      in  vocab_size (crslab.model.crs.kgsf.kgsf.KGSFModel
      crslab.model.conversation.transformer.transformer), 26      attribute), 31
                                         in  vocab_size (crslab.model.crs.redial.redial_conv.ReDialConvModel
                                         attribute), 35
                                         vocab_size (crslab.model.policy.mgcg.mgcg.MGCModel
                                         attribute), 41
transpose_for_scores ()      (crslab.model.recommendation.sasrec.modules.SelfAttention
                               method), 50
truncate () (in module crslab.data.dataloader.utils),
             15

```

## U

```

unk_token_idx (crslab.model.crs.redial.redial_conv.ReDialConvModel
               attribute), 35
use_dropout (crslab.model.crs.redial.redial_conv.ReDialConvModel
               attribute), 36
user_emb_dim (crslab.model.crs.kbrd.kbrd.KBRDModel
               attribute), 29
user_proj_dim (crslab.model.crs.kbrd.kbrd.KBRDModel
               attribute), 29
utterance_encoder_hidden_size
               (crslab.model.crs.redial.redial_conv.ReDialConvModel
               attribute), 36

```

## V

```

valid_data (crslab.data.dataset.durecdial.durecdial.DuRecDialDataset
            attribute), 15
valid_data (crslab.data.dataset.gorecdial.gorecdial.GoRecDialDataset
            attribute), 16
valid_data (crslab.data.dataset.inspired.inspired.InspiredDataset
            attribute), 18
valid_data (crslab.data.dataset.opendialkg.opendialkg.OpenDialKGDataset
            attribute), 19
valid_data (crslab.data.dataset.redial.redial.ReDialDataset
            attribute), 20
valid_data (crslab.data.dataset.tgredial.tgredial.TGReDialDataset
            attribute), 21
vocab (crslab.data.dataset.durecdial.durecdial.DuRecDialDataset
       attribute), 15
vocab (crslab.data.dataset.gorecdial.gorecdial.GoRecDialDataset
       attribute), 17
vocab (crslab.data.dataset.inspired.inspired.InspiredDataset
       attribute), 18
vocab (crslab.data.dataset.opendialkg.opendialkg.OpenDialKGDataset
       attribute), 19

```