```
from unittest.mock import Mock, patch


import pytest

import torch


from swarm_models import TimmModel



def test_timm_model_init():

    with patch("swarms.models.timm.list_models") as mock_list_models:

        model_name = "resnet18"

        pretrained = True

        in_chans = 3

        timm_model = TimmModel(model_name, pretrained, in_chans)

        mock_list_models.assert_called_once()

        assert timm_model.model_name == model_name

        assert timm_model.pretrained == pretrained

        assert timm_model.in_chans == in_chans

        assert timm_model.models == mock_list_models.return_value



def test_timm_model_call():

    with patch(

        "swarms.models.timm.create_model"

    ) as mock_create_model:

        model_name = "resnet18"
        pretrained = True
        in_chans = 3
        timm_model = TimmModel(model_name, pretrained, in_chans)
        task = torch.rand(1, in_chans, 224, 224)
        result = timm_model(task)
        mock_create_model.assert_called_once_with(
            model_name, pretrained=pretrained, in_chans=in_chans
        )
        assert result == mock_create_model.return_value(task)


def test_timm_model_list_models():
    with patch("swarms.models.timm.list_models") as mock_list_models:
        model_name = "resnet18"
        pretrained = True
        in_chans = 3
        timm_model = TimmModel(model_name, pretrained, in_chans)
        result = timm_model.list_models()
        mock_list_models.assert_called_once()
        assert result == mock_list_models.return_value


def test_get_supported_models():
    model_handler = TimmModel()
    supported_models = model_handler._get_supported_models()
    assert isinstance(supported_models, list)
```

```python
    assert len(supported_models) > 0


def test_create_model(sample_model_info):
    model_handler = TimmModel()
    model = model_handler._create_model(sample_model_info)
    assert isinstance(model, torch.nn.Module)


def test_call(sample_model_info):
    model_handler = TimmModel()
    input_tensor = torch.randn(1, 3, 224, 224)
    output_shape = model_handler.__call__(
        sample_model_info, input_tensor
    )
    assert isinstance(output_shape, torch.Size)


def test_get_supported_models_mock():
    model_handler = TimmModel()
    model_handler._get_supported_models = Mock(
        return_value=["resnet18", "resnet50"]
    )
    supported_models = model_handler._get_supported_models()
    assert supported_models == ["resnet18", "resnet50"]
```

```
def test_create_model_mock(sample_model_info):

    model_handler = TimmModel()

    model_handler._create_model = Mock(return_value=torch.nn.Module())

    model = model_handler._create_model(sample_model_info)

    assert isinstance(model, torch.nn.Module)



def test_coverage_report():

    # Install pytest-cov

    # Run tests with coverage report

    pytest.main(["--cov=my_module", "--cov-report=html"])
```