# Contributing

Thanks for considering contributing! We want AllenNLP to be \*the way\* to do cutting-edge NLP research, but we cannot

get there without community support.

## How Can I Contribute?

### Bug fixes and new features

\*\*Did you find a bug?\*\*

First, do [a quick search](https://github.com/allenai/allennlp/issues) to see whether your issue has already been reported.

If your issue has already been reported, please comment on the existing issue.

Otherwise, open [a new GitHub issue](https://github.com/allenai/allennlp/issues). Be sure to include a clear title

and description. The description should include as much relevant information as possible. The description should

explain how to reproduce the erroneous behavior as well as the behavior you expect to see. Ideally you would include a

code sample or an executable test case demonstrating the expected behavior.

\*\*Do you have a suggestion for an enhancement?\*\*

We use GitHub issues to track enhancement requests. Before you create an enhancement request:

\* Make sure you have a clear idea of the enhancement you would like. If you have a vague idea, consider discussing

it first on a GitHub issue.

\* Check the documentation to make sure your feature does not already exist.

\* Do [a quick search](https://github.com/allenai/allennlp/issues) to see whether your enhancement has already been suggested.

When creating your enhancement request, please:

\* Provide a clear title and description.

\* Explain why the enhancement would be useful. It may be helpful to highlight the feature in other libraries.

\* Include code examples to demonstrate how the enhancement would be used.

### Making a pull request

When you're ready to contribute code to address an open issue, please follow these guidelines to help us be able to review your pull request (PR) quickly.

1. \*\*Initial setup\*\* (only do this once)

<details><summary>Expand details ?</summary><br/>

If you haven't already done so, please [fork](https://help.github.com/en/enterprise/2.13/user/articles/fork-a-repo) this repository on GitHub.

Then clone your fork locally with

git clone https://github.com/USERNAME/allennlp.git

or

git clone git@github.com:USERNAME/allennlp.git

At this point the local clone of your fork only knows that it came from \*your\* repo, github.com/USERNAME/allennlp.git, but doesn't know anything the \*main\* repo, [https://github.com/allenai/allennlp.git](https://github.com/allenai/allennlp). You can see this by running

git remote -v

which will output something like this:

origin https://github.com/USERNAME/allennlp.git (fetch)

origin https://github.com/USERNAME/allennlp.git (push)

This means that your local clone can only track changes from your fork, but not from the main repo, and so you won't be able to keep your fork up-to-date with the main repo over time. Therefor you'll need to add another "remote" to your clone that points to [https://github.com/allenai/allennlp.git](https://github.com/allenai/allennlp). To do this, run the following:

git remote add upstream https://github.com/allenai/allennlp.git

Now if you do `git remote -v` again, you'll see

origin https://github.com/USERNAME/allennlp.git (fetch)

origin https://github.com/USERNAME/allennlp.git (push)

upstream https://github.com/allenai/allennlp.git (fetch)

upstream https://github.com/allenai/allennlp.git (push)

Finally, you'll need to create a Python 3.6 or 3.7 virtual environment suitable for working on AllenNLP. There a number of tools out there that making working with virtual environments easier, but the most direct way is with the [`venv` module](https://docs.python.org/3.7/library/venv.html) in the standard library.

Once your virtual environment is activated, you can install your local clone in "editable mode" with

pip install -e .

pip install -r dev-requirements.txt

The "editable mode" comes from the `-e` argument to `pip`, and essential just creates a symbolic link from the site-packages directory of your virtual environment to the source code in your local clone. That way any changes you make will be immediately reflected in your virtual environment.

</details>

2. \*\*Ensure your fork is up-to-date\*\*

<details><summary>Expand details ?</summary><br/>

Once you've added an "upstream" remote pointing to [https://github.com/allenai/allennlp.git](https://github.com/allenai/allennlp), keeping your fork up-to-date is easy:

git checkout master # if not already on master

git pull --rebase upstream master

git push

</details>

3. \*\*Create a new branch to work on your fix or enhancement\*\*

<details><summary>Expand details ?</summary><br/>

Commiting directly to the master branch of your fork is not recommended. It will be easier to keep your fork clean if you work on a seperate branch for each contribution you intend to make.

You can create a new branch with

# replace BRANCH with whatever name you want to give it

git checkout -b BRANCH

git push -u origin BRANCH

</details>

4. \*\*Test your changes\*\*

<details><summary>Expand details ?</summary><br/>

Our continuous integration (CI) testing runs [a number of checks](https://github.com/allenai/allennlp/actions?query=workflow%3APR) for each pull request on [GitHub Actions](https://github.com/features/actions). You can run most of these tests locally, which is something you should do \*before\* opening a PR to help speed up the review process and make it easier for us.

First, you should run [`black`](https://github.com/psf/black) to make sure you code is formatted consistently. Many IDEs support code formatters as plugins, so you may be able to setup black to run automatically everytime you save. [`black.vim`](https://github.com/psf/black/tree/master/plugin) will give you this functionality in Vim, for example. But `black` is also easy to run directly from the command line. Just run this from the root of your clone:

black .

Our CI also uses [`flake8`](https://github.com/allenai/allennlp/tree/master/tests) to lint the code base and [`mypy`](http://mypy-lang.org/) for type-checking. You should run both of these next with

make lint

and

make typecheck

We also strive to maintain high test coverage, so most contributions should include additions to [the unit tests](https://github.com/allenai/allennlp/tree/master/tests). These tests are run with [`pytest`](https://docs.pytest.org/en/latest/), which you can use to locally run any test modules that you've added or changed.

For example, if you've fixed a bug in `allennlp/nn/util.py`, you can run the tests specific to that module with

pytest -v tests/nn/util\_test.py

Our CI will automatically check that test coverage stays above a certain threshold (around 90%). To check the coverage locally in this example, you could run

pytest -v --cov allennlp.nn.util tests/nn/util\_test.py

If your contribution involves changes to any docstrings, you should make sure the API documentation can build without errors. For that, just run

make build-docs

If the build fails, it's most likely due to small formatting issues. If the error message isn't clear, feel free to comment on this in your pull request.

You can also serve and view the docs locally with

make serve-docs

And finally, please update the [CHANGELOG](https://github.com/allenai/allennlp/blob/master/CHANGELOG.md) with notes on your contribution in the "Unreleased" section at the top.

After all of the above checks have passed, you can now open [a new GitHub pull request](https://github.com/allenai/allennlp/pulls).

Make sure you have a clear description of the problem and the solution, and include a link to relevant issues.

We look forward to reviewing your PR!

</details>

### New models

\*\*Do you have a new state-of-the-art model?\*\*

We are always looking for new models to add to our collection. The most popular models are usually added to the official [AllenNLP Models](https://github.com/allenai/allennlp-models) repository, and in some cases to the [AllenNLP Demo](https://demo.allennlp.org/).

If you think your model should be part of AllenNLP Models, please [create a pull request](https://github.com/allenai/allennlp-models/pulls) in the models repo that includes:

\* Any code changes needed to support your new model.

\* A link to the model itself. Please do not check your model into the GitHub repository, but instead upload it in the

PR conversation or provide a link to it at an external location.

In the description of your PR, please clearly explain the task your model performs along with the relevant metrics on an established dataset.