1. **Convert .PT file into ONNX Model**
2. **Modified ONNX model**
3. **Convert ONNX into RKNN Model**
4. **Convert .PT file into ONNX Model**

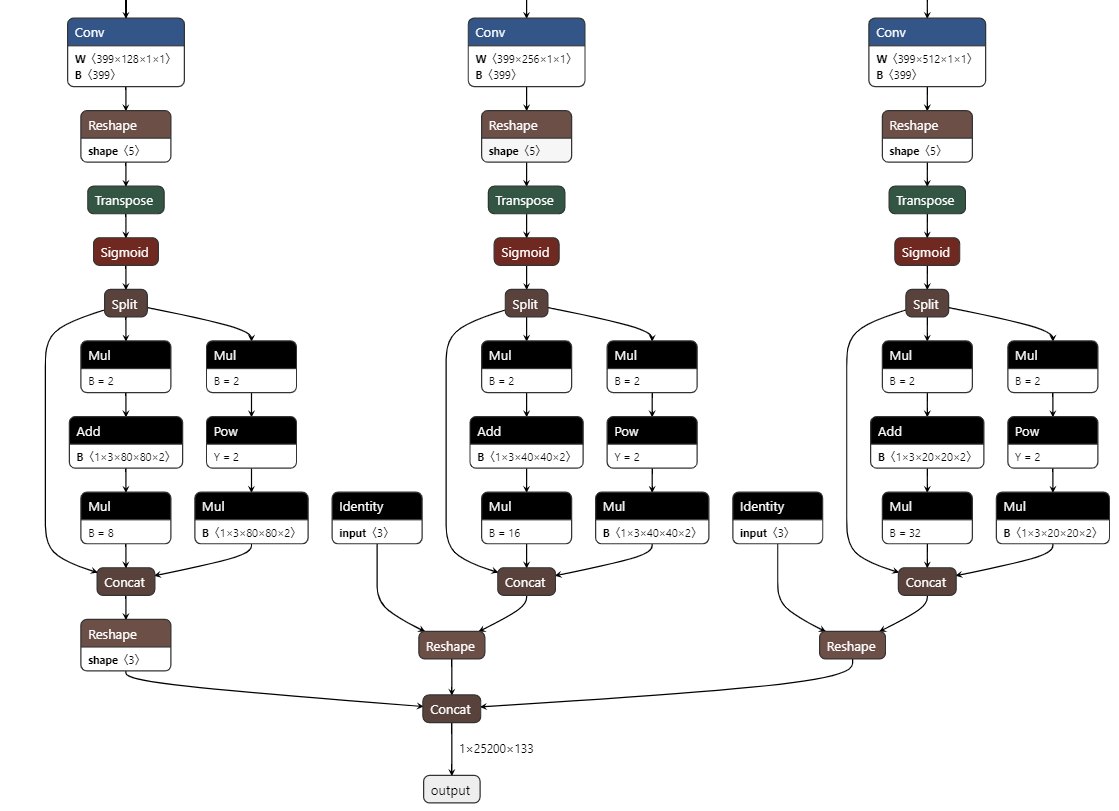
**python export.py --weights gas\_10.26.pt --include onnx**

To convert .PT model into ONNX model we will use above command line in Yolov5 and it will generate ONNX model.

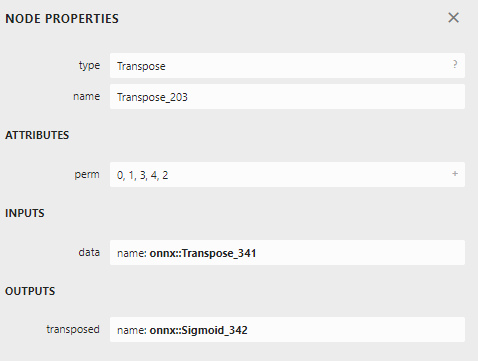
So now we have ONNX model (**gas\_10.26.onnx**)

Open <https://netron.app> to view the onnx model structure

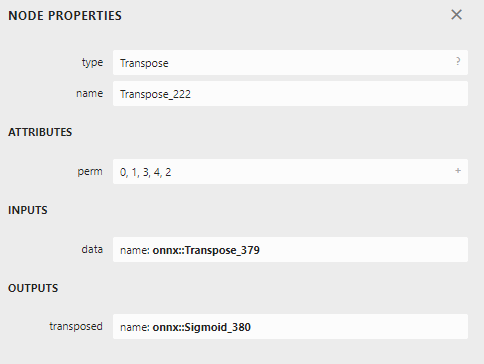
Open **gas\_10.26.onnx** model within <https://netron.app> to see the model structure



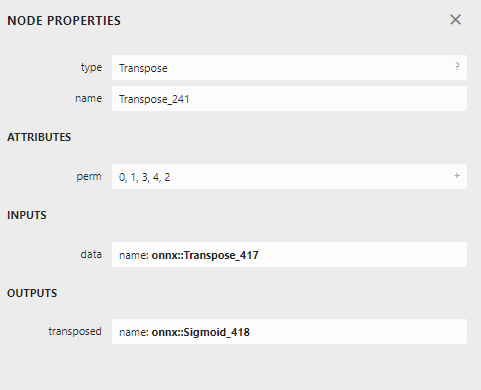
* **Transpose Node 1 Properties at 128**



* **Transpose Node 2 Properties at 256**



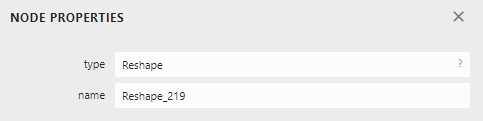
* **Transpose Node 3 Properties at 512**



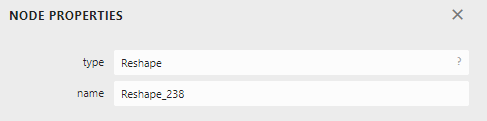
**Concat node properties:**



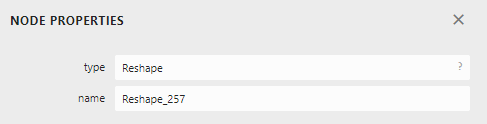
**Reshape Node 1 properties:**



**Reshape Node 2 properties:**



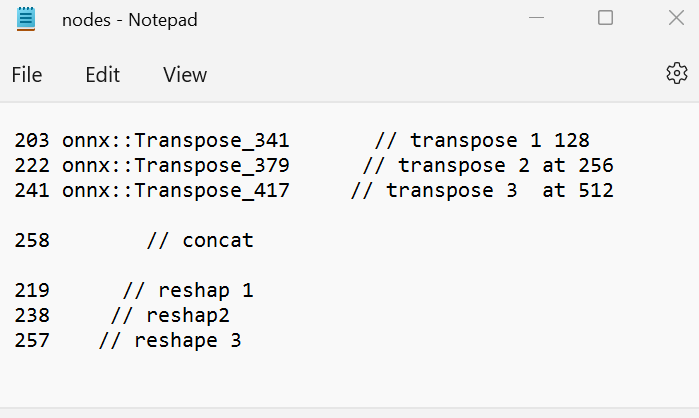
**Reshape Node 3 properties:**



1. **Modified ONNX model**

To convert ONNX model into RKNN we need to modify the generated onnx model into modified one.

So for the we need to note down the transpose three transpose nodes and three reshape nodes and one concat node as mentioned in below image:



Download Android SDK from GitLab and open rknn-toolkit2-1.3.0 to make changes in onxx\_change.py file in below directories:

Downloads/android\_sdk-master/android\_sdk-master/rknn-toolkit2-1.3.0/onnx\_change.py

* **Give path directory to the onnx model (gas\_10.26.onnx)**

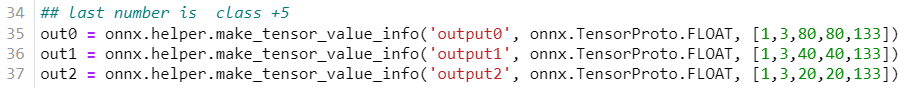


* **Set the concat name value**

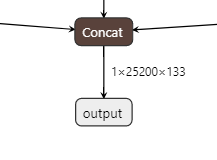
**Our model’s concat name value is 258**

****

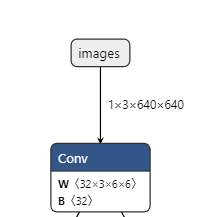
* **Change in output image Matrix**

****

**“133” is the last number which we get from netron.app**

****

**“80” we get after 640 divided by 8. And 640 is our input image in the model which we can see in the netron.app model structure. 40 we get divide 80 by 2 and then divide 40 by 2 to get 20. Now our matrix is ready.**

****

* **Change in below code line with respect to the below nodes values:**

1. **Transpose**
2. **Cancat**
3. **Reshape**

****

**After changes run** onxx\_change.py file to generate modified onnx model. Structure of modifies onxx model we can see on netron.app. Now time to generate RKNN model from modifies onnx model.

1. **Convert ONNX into RKKN Model**

**Open below directories:**

**Downloads/android\_sdk-master/android\_sdk-master/rknn-toolkit2-1.3.0/examples/onnx/yolov5/test.py**

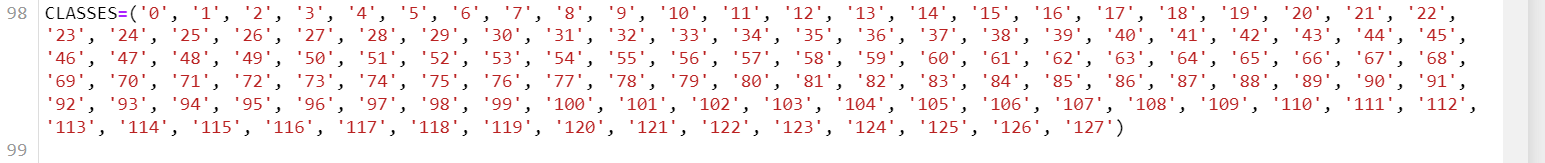
**Make below changes in test.py model and run test.py to generate RKNN model weight file which we will use in RK3588 for writing C++ code and generate APK file.**

**Add to path:**

****

**Add class lables:**

**We have total 128 class labels.**

****

**For convenience we have given labels as number but for real testing we need to give real label names:**

**Everything is done now run the test.py to get RKNN model. Our real class label which we can see in netron.app:**

****

**Next step is to test image detection on RKNN model and writing C++ logic code.**