Item 311
git_comments:

1. astype since numpy functions default return type is boolean array instead of int

git_commits:

1. **summary:** [MXNET-1173] Debug operators - isfinite, isinf and isnan (#12967)
   message: [MXNET-1173] Debug operators - isfinite, isinf and isnan (#12967) * is_finite and is_inf implementation for front-end python api debug operator * updated unit-tests * updated test cases and incorporated is_nan function * solved index out of bounds issue and added comments * simplified abs function call and added isnan to contrib.py and all debug ops to doc * changed dimensions, added regular number, assert_equal instead of almost, removed ctx and added data.abs

github_issues:

1. **title:** Feature Request NDArray.has_nan() method
   body: `ndarray.has_nan(NDArray)` and its convenient form `NDArray.has_nan()`, which return `True` if there is `nan` in the data else return `False`. This is very helpful in debugging. Optimally has an option to output the per location prediction `ndarray.has_nan(x, gather=True)`, when `gather=False`, it should output a binary NDArray with the same shape as `x` and each location predicts the corresponding data is `nan` or not.
2. **title:** Feature Request NDArray.has_nan() method
   body: `ndarray.has_nan(NDArray)` and its convenient form `NDArray.has_nan()`, which return `True` if there is `nan` in the data else return `False`. This is very helpful in debugging. Optimally has an option to output the per location prediction `ndarray.has_nan(x, gather=True)`, when `gather=False`, it should output a binary NDArray with the same shape as `x` and each location predicts the corresponding data is `nan` or not.
3. **title:** Feature Request NDArray.has_nan() method
   body: `ndarray.has_nan(NDArray)` and its convenient form `NDArray.has_nan()`, which return `True` if there is `nan` in the data else return `False`. This is very helpful in debugging. Optimally has an option to output the per location prediction `ndarray.has_nan(x, gather=True)`, when `gather=False`, it should output a binary NDArray with the same shape as `x` and each location predicts the corresponding data is `nan` or not.
4. **title:** Feature Request NDArray.has_nan() method
   body: `ndarray.has_nan(NDArray)` and its convenient form `NDArray.has_nan()`, which return `True` if there is `nan` in the data else return `False`. This is very helpful in debugging. Optimally has an option to output the per location prediction `ndarray.has_nan(x, gather=True)`, when `gather=False`, it should output a binary NDArray with the same shape as `x` and each location predicts the corresponding data is `nan` or not.
5. **title:** Feature Request NDArray.has_nan() method
   body: `ndarray.has_nan(NDArray)` and its convenient form `NDArray.has_nan()`, which return `True` if there is `nan` in the data else return `False`. This is very helpful in debugging. Optimally has an option to output the per location prediction `ndarray.has_nan(x, gather=True)`, when `gather=False`, it should output a binary NDArray with the same shape as `x` and each location predicts the corresponding data is `nan` or not.

github_issues_comments:

1. @mxnet-label-bot [Feature Request, Operator]
2. Similarly NDArray.is_inf() https://www.tensorflow.org/api_docs/python/tf/is_inf
3. Just went through the Debug operators and looks like, there are quite of few of them https://www.tensorflow.org/api_guides/python/control_flow_ops#Debugging_Operations ``` 1. tf.is_finite 2. tf.is_inf 3. tf.is_nan 4. tf.verify_tensor_all_finite 5. tf.check_numerics 6. tf.add_check_numerics_ops 7. tf.Assert 8. tf.Print ``` Mostly I will work on that.

github_pulls:

1. **title:** [MXNET-1173] Debug operators - isfinite, isinf and isnan

body: ## Description ## First 3 Debug operators. Fixes #12623 `isinf`, `isnan` and `isfinite` ## Checklist ## ### Essentials ### Please feel free to remove inapplicable items for your PR. - [X] The PR title starts with [MXNET-$JIRA_ID], where $JIRA_ID refers to the relevant [JIRA issue] (https://issues.apache.org/jira/projects/MXNET/issues) created (except PRs with tiny changes) - [X] Changes are complete (i.e. I finished coding on this PR) - [X] All changes have test coverage: - Unit tests are added for small changes to verify correctness (e.g. adding a new operator) - Nightly tests are added for complicated/long-running ones (e.g. changing distributed kvstore) - Build tests will be added for build configuration changes (e.g. adding a new build option with NCCL) - [X] Code is well-documented: - For user-facing API changes, API doc string has been updated. - For new C++ functions in header files, their functionalities and arguments are documented. - For new examples, README.md is added to explain the what the example does, the source of the dataset, expected performance on test set and reference to the original paper if applicable - Check the API doc at http://mxnet-ci-doc.s3-accelerate.dualstack.amazonaws.com/PR-$PR_ID/$BUILD_ID/index.html - [ ] To the my best knowledge, examples are either not affected by this change, or have been fixed to be compatible with this change ### Changes ### - [ ] Added unit tests - [ ] Added functions in python front-end api ## Comments ## - just the 2 of the total 6-7 operators that can be clubbed under "debug operators"

github_pulls_comments:

1. @ChaiBapchya Thanks for the contribution! @mxnet-label-bot [pr-awaiting-review]

github_pulls_reviews:

1. the description should include the word element-wise. Something like, "Performs an element-wise check to determine if the NDArray contains an infinite element or not"
2. This should specify that the output has the same shape as input, and that it will be 1 where the array element is equal to +/- infinity and is 0 in other places.
3. same comments as above.
4. make a check for `Nan` values
5. this method should also check for `Nan` values and return 0 in places where the element == `Nan`
6. randomize the shape and size of the array in this test case
7. **body:** do not hardcode the expected output. Calculate it using numpy.
   label: code-design
8. It returns 0 for NaN even now.
9. for `isinf` it was returning 0 for NaN (the way it was written) however for `is_finite` it wasn't returning 0 so now worked on it. Thanks ultimately both `isinf` and `isfinite` return 0 for NaN
10. Thanks for that.
11. Could you also update https://github.com/apache/incubator-mxnet/blob/master/docs/api/python/ndarray/contrib.md#contrib ?
12. simply use `data.abs() == np.inf`?
13. is ctx taking effect and tested? Why are we adding the ctx option to these ops?
14. Sure. simpler. Thanks
15. In case ctx isn't passed, hence I'm checking for that and setting. In CI tests, it passes for CPU as well as GPU, so I am guessing it's fine. Am I missing something?
16. Done
17. Same here: use data.abs()
18. **body:** same here, why ctx is passed in while not used.
    label: code-design
19. also add regular number in the data array
20. since you create data with random dimensions, but why only set number in three dimensions?
21. same here why only set three dimensions while you create data with random dimensions.
22. this should be assert_equal not assert_almost_equal
23. this should be assert_equal not assert_almost_equal
24. This was after discussion with @anirudhacharya about how to verify that cases of infinite, negative infinite and NaN are handled.
25. As an after thought, nothing context specific is being done. So removing it.
26. As an after thought, nothing context specific is being done. So removing it.
27. the function - `rand_ndarray` in `/python/mxnet/test_utils` is meant to do exactly that - initialize the array with random numbers

28. Anyway, incorporated that nonetheless.
29. Makes sense. Since I am passing the same data, it is necessary that it's exactly equal (not "almost") Thanks for pointing that out.
30. the dimensions for setting of nan and np.inf values can be randomized too.

jira_issues:

1. **summary:** Debug Operator isfinite and isinf
   description:

jira_issues_comments: