Item 292
git_comments:

1. ! \brief input is one tensor
2. For Sparse CSR
3. ! \brief Binary launch

git_commits:

1. **summary:** [numpy] change unary infer type (#17254)
   message: [numpy] change unary infer type (#17254) * cpu ok * gpu ok * change name * sanity * re-trigger

github_issues:

github_issues_comments:

github_pulls:

1. **title:** [numpy] change unary infer type
   body: ## Description ## change unary infer type ## Checklist ## ### Essentials ### Please feel free to remove inapplicable items for your PR. - [ ] The PR title starts with [MXNET-$JIRA_ID], where $JIRA_ID refers to the relevant [JIRA issue](https://issues.apache.org/jira/projects/MXNET/issues) created (except PRs with tiny changes) - [ ] Changes are complete (i.e. I finished coding on this PR) - [ ] All changes have test coverage: - Unit tests are added for small changes to verify correctness (e.g. adding a new operator) - Nightly tests are added for complicated/long-running ones (e.g. changing distributed kvstore) - Build tests will be added for build configuration changes (e.g. adding a new build option with NCCL) - [ ] Code is well-documented: - For user-facing API changes, API doc string has been updated. - For new C++ functions in header files, their functionalities and arguments are documented. - For new examples, README.md is added to explain the what the example does, the source of the dataset, expected performance on test set and reference to the original paper if applicable - Check the API doc at https://mxnet-ci-doc.s3-accelerate.dualstack.amazonaws.com/PR-$PR_ID/$BUILD_ID/index.html - [ ] To the best of my knowledge, examples are either not affected by this change, or have been fixed to be compatible with this change ### Changes ### - [ ] Feature1, tests, (and when applicable, API doc) - [ ] Feature2, tests, (and when applicable, API doc) ## Comments ## - If this change is a backward incompatible change, why must this change be made. - Interesting edge cases to note here

github_pulls_comments:

1. @mxnet-bot run ci [unix-gpu]
2. Jenkins CI successfully triggered : [unix-gpu]
3. @mxnet-bot run ci [centos-gpu]
4. Jenkins CI successfully triggered : [centos-gpu]

github_pulls_reviews:

1. please get a better name.
2. better use `OType` and `IType`
3. also, better name for this one.
4. same here, get a better name for this.
5. better name for both `Compute2` and `Compute3`
6. remove dead code.
7. 1 more blank line above.
8. get rid of this blank line.
9. Change the name to `TestMixedUnary`
10. I think both `rad2deg` and `deg2rad` support mixed precision.
11. Please also add checks for backward computation for floating point input cases.
12. Simply use `np.array` instead of `mx.numpy.array`
13. `MixedUnaryOpType`

14. pay attention to alignments
15. use `kFloat32` as default for now.
16. `MXNET_OPERATOR_REGISTER_NUMPY_MIXED_TYPE_UNARY`
17. `MXNET_OPERATOR_REGISTER_NUMPY_MIXED_TYPE_UNARY_GPU `
18. alignment
19. alignment
20. `MixedUnaryBackwardUseInCompute`
21. `MixedUnaryBackwardUseInOutCompute`
22. use `MXNET_REAL_TYPE_SWITCH` here since you already excluded the integer case.
23. I think you should directly check if the `output[0]`'s type is an integer type here.
24. alignment
25. `MSHADOW_REAL_TYPE_SWITCH` here since you've masked out integer cases above.
26. alignment
27. this is the case when forward computation's input is floating point number, the output will be of the same type as the input, so you can directly call the original `Compute` function above, no need for launching the kernel here.
28. this case and the above case could be merged. Plus, the error message is not meaningful enough. `mshadow::dtype_string(outputs[0].type_flag_)` will give you the corresponding string that represents the data type. Also the problem with showing the op's name is that it will be something in the form of `_backward_npi_xxx`. I think here simply say that "gradient computation for xxx type is not supported" is better. ("xxx" should be the string returned by call to `dtype_string`)
29. Same for the error message here since `outputs[0].type_flag_` is technically same as `inputs[1].type_flag_`
30. no need for `WITH_SPARSE` since this does not include sparse functionalities.
31. `MXNET_OPERATOR_REGISTER_UNARY_MIXEDTYPE_USEIN_BWD_CPU`
32. `MXNET_OPERATOR_REGISTER_UNARY_MIXEDTYPE_USEINOUT_BWD_CPU`
33. `MXNET_OPERATOR_REGISTER_UNARY_MIXEDTYPE_BWD_IN` here and `MXNET_OPERATOR_REGISTER_UNARY_MIXEDTYPE_BWD_INOUT` below
34. seems like `FResourceRequest` is not used, you can simply get rid of this `set_attr`

jira_issues:

jira_issues_comments: