INVERSE DESIGN OF MATERIALS THAT EXHIBIT THE MAGNETOCALORIC EFFECT BY TEXT-MINING OF THE SCIENTIFIC LITERATURE AND GENERATIVE DEEP LEARNING

Supporting Information

Callum J. Court[1], **Apoorv Jain**[1,2], and **Jacqueline M. Cole**[1,2,3 *]

[1] *Cavendish Laboratory, Department of Physics, University of Cambridge, J. J. Thomson Avenue, Cambridge, CB3 0HE, UK*

[2] *Department of Chemical Engineering and Biotechnology, University of Cambridge, West Cambridge Site, Philippa Fawcett Drive, Cambridge, CB3 0FS, UK*

[3] *ISIS Neutron and Muon Source, STFC Rutherford Appleton Laboratory, Harwell Science and Innovation Campus, Didcot, OX11 0QX, UK*

[*]Corresponding Author: `jmc61@cam.ac.uk`

August 29, 2021

1 ChemDataExtractor 2.0 models

Here we provide the source code for the creation of the `CurieTemperature`, `MagneticEntropy`, `RelativeCoolingPower` and `MagneticField` models.

```python
from chemdataextractor.model import BaseModel, ListType, ModelType, StringType
from chemdataextractor.model.units.temperature import TemperatureModel
from chemdataextractor.model.units.energy import Energy
from chemdataextractor.model.units.mass import Mass
from chemdataextractor.parse.actions import join
from chemdataextractor.parse.auto import AutoTableParser
from chemdataextractor.parse.common import dt, lbrct, rbrct
from chemdataextractor.parse.elements import I, Optional, R, W, ZeroOrMore
from chemdataextractor.parse.template import (MultiQuantityModelTemplateParser,
                                              QuantityModelTemplateParser)
from .inorganic_compound import InorganicCompound, inorganic_names_only
from .magnetic_field import MagneticField

delim = R("^[:;\.,]$")

# Curie Temperature Specifier
curie_specifier = (
    (
        R("^\[?T(C|c)(urie)?[1-2]?\]?$")
        | R("^T(O|o)rd$")
        | R("^T(M|M)ag$")
        | R("^(C|c)urie$")
        | R("^Tt$")
        | R("^(F|f)err[io]magnet(ic|ism|ically)?$")
        | I("FM")
        | I("ferroelectric")
        | W("(PM)-ferromagnetic")
    )
    + Optional(lbrct + I("FM") + rbrct).hide()
    + Optional(
        I("ordering")
        | I("ordered")
        | I("order-disorder")
        | I("behavior")
        | I("behaviour")
        | I("order")
    )
    + Optional(I("phase"))
    + Optional(R("^transition(s)?$"))
    + Optional(R("^temperature[s]?$") |
               R("^temp[\.]?$") | I("point") | I("value"))
    + ZeroOrMore(delim | lbrct).hide()
    + Optional(I("at") | I("with")).hide()
    + Optional(
        R("^\[?T(C|c)(urie)?[1-2]?\]?$") | I("TM") | R("^T(O|o)rd$")
```

```
            | R("^T(M|M)ag$")
    )
    + ZeroOrMore(delim | rbrct).hide()
).add_action(join)

# Magnetic Entropy Specifier
magent_specifier = (
    ((Optional(I("magnetic") | I("isothermal")) + I("entropy") + I("change"))
        | (R("^\|?[\-\−]?\(?[ΔΔΔ]S([Mm]\)?(m?ax)?|peak|pk)?\|?(pk)?")
           + Optional(R("[Mm]ax")))
        | (R("^[ΔΔΔ]S$") + R("[Mm](ax)?"))
    )
    + ZeroOrMore(delim | lbrct).hide()
    + Optional(I("at") | I("with") | I("of")).hide()
    + Optional(
        (R("^\|?[\-\−]?\(?[ΔΔ]S([Mm]\)?(m?ax)?|peak)?\|?(pk)?")
            + Optional(R("[Mm]ax")))
        | (R("^[ΔΔΔ]S$") + R("[Mm](ax)?"))
    )
    + ZeroOrMore(delim | rbrct).hide()
).add_action(join)

# Relative Cooling Power Specifier
rcp_specifier = (
    (
        (I("relative") + I("cooling") + I("power"))
        | ((I("refrigeration") | I("refrigerant")) + I("capacity"))
        | R("RC((P)?(\(?FWHM\)?)?)?")
    )
    + ZeroOrMore(delim | lbrct).hide()
    + Optional(I("at") | I("with")).hide()
    + Optional(R("RC((P)?(\(?FWHM\)?)?)?"))
    + ZeroOrMore(delim | rbrct).hide()
).add_action(join)

# Magnetic Field Specifier
magnetic_field_specifier = (
    (
        R("μ0H(app)?")
        | (R("μ0") + W("H"))
        | (R("μ") + R("0") + W("H"))
        | R("μ0?[ΔΔ]H(app)?")
        | (R("μ0") + R("[ΔΔ]H(app)?"))
        | R("[ΔΔΔ]H(app)?")
        | R("H[Mm]ax")
```

```
        | ((I("magnetic") | I("applied")) + I("field")
              + Optional(I("change")))
    )
    + ZeroOrMore(delim | lbrct).hide()
    + Optional(I("at") | I("with")).hide()
    + Optional(
        R("μ0H")
        | (R("μ0") + W("H"))
        | (R("μ") + R("0") + W("H"))
        | R("μ0[ΔΔ]H(app)?")
        | (R("μ0") + R("[ΔΔ]H(app)?"))
    )
    + ZeroOrMore(delim | rbrct).hide()
).add_action(join)

"""Curie Temperature Model

Dimensions: Temperature
"""
class CurieTemperature(TemperatureModel):
    specifier = StringType(
        parse_expression=curie_specifier,
        required=True,
        contextual=True,
    )
    compound = ModelType(
        InorganicCompound,
           required=True,
           contextual=True
    )
    parsers = [
        MultiQuantityModelTemplateParser(),
        QuantityModelTemplateParser(),
        AutoTableParser(chem_name=inorganic_names_only),
    ]

"""Magnetic Entropy model

Dimensions: Energy() / (Mass() * Temperature())
"""

class MagneticEntropy(QuantityModel):
    specifier = StringType(
        parse_expression=magent_specifier,
           required=True,
```

```
            contextual=True,
            updatable=True
    )
    compound = ModelType(
        InorganicCompound,
            required=True,
            contextual=True,
            updatable=False
    )
    field = ModelType(
            MagneticField,
            required=True,
            contextual=True,
            updatable=False)
    dimensions = Energy() / (Mass() * Temperature())
    parsers = [
        MultiQuantityModelTemplateParser(),
        QuantityModelTemplateParser(),
        AutoTableParser(),
    ]

"""Relative Cooling Power model

Dimensions: Energy() / Mass()
"""
class RelativeCoolingPower(QuantityModel):
    specifier = StringType(
        parse_expression=specifier,
            required=True,
            contextual=True,
            updatable=True
    )

    compound = ModelType(
        InorganicCompound,
            required=True,
            contextual=True,
            updatable=False
    )

    field = ModelType(
            MagneticField,
            required=True,
            contextual=True,
            updatable=False)
```

```

    dimensions = Energy() / Mass()
    parsers = [
        MultiQuantityModelTemplateParser(),
        QuantityModelTemplateParser(),
        AutoTableParser(chem_name=inorganic_names_only),
    ]

"""Magnetic Field Model
Dimensions: MagneticField()

"""
class MagneticField(MagneticFieldModel):
    specifier = StringType(
        parse_expression=magnetic_field_specifier,
        required=True,
        contextual=True,
        updatable=True,
    )
    compound = ModelType(
        InorganicCompound,
           required=True,
           contextual=True,
           updatable=False
    )
    parsers = [
        AutoSentenceParser(chem_name=inorganic_names_only),
        AutoTableParser(chem_name=inorganic_names_only),
    ]
```

2 Magnetocaloric Regression Model Features

The table below provides a list of the 58 element-level features and the magnetic field feature used to train the magnetocaloric property prediction models. All element-level features were obtained from the XenonPy python library[?] .

Each compound is converted to seven composition-weighted feature vectors, using the weighted average, weighted sum, weighted variance, maximum pooling, minimum pooling, geometric mean and harmonic mean. For example, consider a binary compound, $A_{w_A}B_{w_B}$ with element-level features denoted by $f_{A,i}$ and $f_{B,i}$ for $i = (1,...,58)$. We then compute the feature descriptors as

$$\text{Weighted average} = w_A^* f_{A,i} + w_B^* f_{B,i} \tag{1}$$

$$\text{Weighted sum} = w_A f_{A,i} + w_B f_{B,i} \tag{2}$$

$$\text{Weighted variance} = w_A^*(f_{A,i} - f_{ave,i})^2 + w_B^*(f_{B,i} - f_{ave,i})^2 \tag{3}$$

$$\text{Geometric mean} = \sqrt[w_A+w_B]{f_{A,i}^{w_A} * f_{V,i}^{w_B}} \tag{4}$$

$$\text{Harmonic mean} = \frac{w_A + w_B}{\frac{1}{f_{A,i}} * w_A + \frac{1}{f_{B,i}} * w_B} \tag{5}$$

$$\text{Maximum pooling} = max f_{A,i}, f_{B,i} \tag{6}$$

$$\text{Minimum pooling} = min f_{A,i}, f_{B,i} \tag{7}$$

Table 1: Element-level feature vectors for the magnetocaloric property prediction models.

Feature	Description
period	Period in the periodic table
atomic_number	Number of protons found in the nucleus of an atom
mendeleev_number	Atom number in Mendeleevs periodic table
atomic_radius	Atomic radius
atomic_radius_rahm	Atomic radius by Rahm et al
atomic_volume	Atomic volume
atomic_weight	The mass of an atom
icsd_volume	Atom volume in ICSD database
lattice_constant	Physical dimension of unit cells in a crystal lattice
vdw_radius	Van der Waals radius
vdw_radius_alvarez	Van der Waals radius according to Alvarez
vdw_radius_mm3	Van der Waals radius from the MM3 FF
vdw_radius_uff	Van der Waals radius from the UFF
covalent_radius_cordero	Covalent radius by Cerdero et al
covalent_radius_pyykko	Single bond covalent radius by Pyykko et al
covalent_radius_pyykko_double	Double bond covalent radius by Pyykko et al
covalent_radius_pyykko_triple	Triple bond covalent radius by Pyykko et al
covalent_radius_slater	Covalent radius by Slater
c6_gb	C_6 dispersion coefficient in a.u
density	Density at T = 295 K
dipole_polarizability	Dipole polarizability
electron_affinity	Electron affinity
electron_negativity	Tendency to attract a shared pair of electrons
en_allen	Allen's scale of electronegativity
en_ghosh	Ghosh's scale of electronegativity
en_pauling	Mulliken's scale of electronegativity
gs_bandgap	DFT bandgap energy of T = 0 K ground state
gs_energy	DFT energy per atom of T = 0 K ground state
gs_est_bcc_latcnt	Estimated BCC lattice parameter
gs_est_fcc_latcnt	Estimated FCC lattice parameter
gs_mag_moment	DFT magnetic moment of T = 0 K ground state
gs_volume_per	DFT volume per atom of T = 0 K ground state
hhi_p	Herfindahl-Hirschman Index production values
hhi_r	Herfindahl-Hirschman Index reserves values
specific_heat	Specific heat at 20°C
first_ion_en	First ionisation energy
fusion_enthalpy	Fusion heat
heat_of_formation	Heat of formation
heat_capacity_mass	Mass specific heat capacity
heat_capacity_molar	Molar specific heat capacity
boiling_point	Boiling temperature
bulk_modulus	Bulk modulus
melting_point	Melting point
thermal_conductivity	Thermal conductivity at 25 °C
sound_velocity	Speed of sound
Polarizability	Ability to form instantaneous dipoles
poissons_ratio	Poissons ratio
molar_volume	Molar volume
num_unfilled	Total unfilled electrons
num_valence	Total valence electrons
num_d_unfilled	Unfilled electrons in d shell
num_d_valence	Valence electrons in d shell
num_f_unfilled	Unfilled electrons in f shell
num_f_valence	Valence electrons in f shell
num_p_unfilled	Unfilled electrons in p shell

Feature	Description
num_p_valence	Valence electrons in p shell
num_s_unfilled	Unfilled electrons in s shell
num_s_valence	Valence electrons in s shell
applied_field	Field measurement for RCP and entropy models